Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft

Wissenschaftliche Berichte

FZKA 7523

Real-Time Mapping of Rotationally Symmetric Objects for Mobile Inspection

P. Kohlhepp

Institut für Angewandte Informatik

Oktober 2009

Forschungszentrum Karlsruhe
in der Helmholtz-Gemeinschaft
Wissenschaftliche Berichte
FZKA 7523

Real-Time Mapping of Rotationally Symmetric Objects for Mobile Inspection

P. Kohlhepp
Institut für Angewandte Informatik

Forschungszentrum Karlsruhe GmbH, Karlsruhe
2009

Forschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

Mitglied der Hermann von Helmholtz-Gemeinschaft
Deutscher Forschungszentren (HGF)

ISSN 0947-8620

urn:nbn:de:0005-075231

Abstract

Lifelong monitoring of industrial installations, buildings, and urban settlements as to their thermal energy efficiency requires new, highly automated procedures, for example quantitative geo-referenced thermography. These tools crucially depend on part-based *geometric* and *semantic data models* of the sites. There is a lack of CAD models today that are truly up-to-date and readily applicable to such *cognitive condition monitoring*. Industrial plants and legacy sites also provide an unsafe, sometimes toxic or inaccessible environment to humans.

This report proposes and analyzes algorithms to estimate parameterized object representations automatically in real time while capturing point clouds. Mobile platforms and low-cost 3D laser scanners are applied offering a limited field of view and limited spatial resolution. One most important and challenging class of objects are pipe ducts and vessels in chemical plants. This report focuses on methods for the extraction and fitting of rotationally symmetric objects from single range views, their aggregation to linear or branched duct structures, and their relational characterization for matching and merging them. From these partial object descriptions contiguous site maps are composed by the Elastic View Graph (EVG) framework. EVG is an image-based framework for 3D/6DoF SLAM treating as 'observations' relational attributed feature graphs, in other words: pre-segmented 3D images.

As to main technical contributions, a new Expectation-Maximization (EM) algorithm is proposed for segmentation and parameter fitting of surfaces-of-revolution (SoR) with piecewise straight axis segments. The E-step implements a Bayesian model of sampling from rotation surfaces. For the M-step, two implemented methods are offered: cone fitting by the geometric distance, implemented by the Levenberg-Marquardt nonlinear optimization algorithm, and an adapted Iterative Closest Point (ICP) algorithm with separately estimated shape parameters. EM converges (empirically) to a *local* maximum of the posterior probability.

Since the optimization space spanning the SoR parameters and the set of point partitions has so many local optima, good starting values are vital. A key focus is therefore on effective hypothesis generation. To this end, three partly new algorithms are developed. The first one, the Chain algorithm, builds a mesh of paths in the directions of principal (Min/Max) curvatures. The second one uses foot point transformations which are local operators contracting a rotation surface to a noisy point representation of its axis curve. By means of an adapted Principal Curve algorithm, this foot point cloud is decomposed into piecewise linear axis segments. The third method exploits the characterizing property of SoR that all normal rays intersect the axis. Using line geometry an axis is estimated in closed form and, by evaluating the degree of rotational symmetry, the point set is recursively subdivided if necessary.

In an extensive case study, all algorithms are evaluated and compared on real images from the experimental thermal plant THERESA at KIT North Campus captured by a rotating 3D laser scanner. Quantitative results include the hypothesis evaluation and decimation of false-positive hypotheses, the assessment and comparison of algorithm stability with respect to various input variations (random sampling, Gaussian noise, unknown observer motion during image capture), and the EM convergence speed. The total estimation times are essentially linear in the image size and keep up with the scanner, i.e. admit real-time operation.

Echtzeit-Kartierung rotationssymmetrischer Anlagenkomponenten zur mobilen Inspektion

Zusammenfassung

Die kontinuierliche Überwachung der thermischen Effizienz von Industrieanlagen, Gebäuden und Siedlungen erfordert neue, hochgradig automatisierte Verfahren, z.B. quantitative georeferenzierte Thermographie. Diese benötigen in Bauteile gegliederte geometrische und semantische Datenmodelle der Anlagen, die aber oft nicht auf dem aktuellen Stand und direkt nutzbar für eine solche *Kognitive Instandhaltung* sind. Für den Menschen stellen Industrieanlagen eine unsichere, manchmal toxische oder unzugängliche Umgebung dar.

Dieser Bericht untersucht neue Algorithmen, um parametrische Datenmodelle aus Punktwolken automatisch und mit deren Erfassung Schritt haltend zu schätzen. Dazu dienen mobile Plattformen und preiswerte 3D Laserscanner, deren Blickfeld und räumliche Auflösung begrenzt sind. Die wichtigste Objektklasse bilden Rohrleitungen, Kessel, Behälter o.ä. in verfahrenstechnischen Anlagen. Im Fokus stehen hier echtzeittaugliche Methoden zur Detektion und Parameter-Einpassung rotationssymmetrischer Körper, die zu Strängen oder Leitungssystemen gruppiert werden und deren Relationen die Zuordnung unterstützen. Aus den Teilmodellen der Einzelansichten wird eine zusammenhängende metrische 3D Karte der Anlage mit Hilfe des "Elastic View Graph" Schemas (EVG) generiert. EVG ist ein bildbasiertes Verfahren für 3D/6DoF SLAM, welches vor-segmentierte 3D-Bilder verwendet.

Ein Hauptbeitrag dieser Arbeit ist ein neuer Expectation-Maximization (EM) Algorithmus zur Parameterschätzung von Rotationsflächen (SoR) mit stückweise gerader Mittellinie. Der E-Schritt implementiert ein Bayes'sches Sensormodell für SoR-Flächen; im M-Schritt kommen zwei alternative Methoden zum Einsatz: entweder nichtlineare Parameteroptimierung nach geometrischer Distanz (Levenberg-Marquardt), oder Optimierung der Lageparameter durch Iterative-Closest-Point (ICP), wobei die Formparameter separat geschätzt werden. EM konvergiert (empirisch) zu einem lokalen Maximum der posteriori Modellwahrscheinlichkeit.

Da der die SoR-Parameter und Bildpartitionen umspannende Optimierungsraum viele lokale Optima besitzt, sind gute Anfangswerte wichtig. Für die effektive Generierung von Hypothesen stehen drei teils neue Algorithmen zur Verfügung. Der erste Algorithmus schätzt ein Netz von Ketten in den Richtungen der Hauptkrümmungen. Die zweite Methode kontrahiert mit einem lokalen Bildoperator die Punkte der Rotationsflächen zu einer verrauschten Mittellinienkurve, die sodann mit Hilfe eines adaptierten Hauptkurvenalgorithmus in stückweise lineare Mittelliniensegmente zerlegt wird. Die dritte Methode nutzt die charakteristische Eigenschaft aus, dass alle Normalstrahlen die SoR-Mittellinie schneiden. Mit Hilfe algebraischer Liniengeometrie wird eine Achse in geschlossener Form geschätzt und abhängig vom Grad der Rotationssymmetrie um die Achse die Punktmenge rekursiv unterteilt.

Alle Algorithmen wurden in einer umfangreichen Fallstudie an realen, mit einem rotierenden 3D Laserscanner erfassten Tiefenbildern aus der thermischen Versuchsanlage THERESA bewertet und verglichen. Quantitativ erfasst wurden die Hypothesen-Bewertung zur Ausdünnung falsch-positiver SoR-Hypothesen, die Stabilität der Algorithmen gegenüber Bildstörungen (zufallsgesteuerte Abtastung, Gauß'sches Rauschen und Bewegungsrauschen bei der Bildaufnahme), sowie die Konvergenzgeschwindigkeit des EM und die Auswirkungen vorzeitiger Terminierung. Die Echtzeitfähigkeit wurde auch überprüft: die Gesamtzeiten der Schätzung sind im wesentlichen linear in der Bildgröße und halten mit der Bildaufnahme Schritt.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

This report describes ongoing work towards a long-term goal: building mobile, highly automated and effective monitoring tools for the routine inspection and predictive maintenance of tomorrow, with a focus on energy efficiency. Their task will be to assess the condition of buildings, settlements, production plants and facilities of public infrastructure, by analyzing thermal heat transfer, material integrity, tightness, and aging processes that may degrade their performance. Facilities will be assessed throughout their life cycle and monitored under operation, non-invasively, using contactless area or volume imaging sensors. Infrared cameras for recording heat distributions (thermography) and FTIR (Fourier-transformation infrared) gas sensors for reconstructing gas concentration maps will play a dominant role.

Collecting huge image databases and filling spatial-temporal grids with measured quantities, thereby creating a possible bottleneck in data evaluation and interpretation, is not the main purpose, however. Rather, topical and on-site understanding is. When taking an image, the monitoring system should instantly and precisely *know* where and to which *part* it belongs, and what the component *function* is. A history of survey data on this part should be quickly accessible. To compare the current image to prior ones may require normalizing the viewing conditions, i.e. to automatically trim and register the images. Not just difference images but useful quantitative figures as the percent in heat loss of insulation material may be what the future maintenance technician desires. To estimate such goal variables requires general laws of radiative heat transfer to be incorporated, but also part-specific knowledge like material properties, ambient conditions, and prior experiences with this part.

Truly efficient tools for condition monitoring must therefore be deeply rooted in *spatial* and *semantic data models* of facilities, and need instant and on-site access to their information. Part based and geo-referenced remote inspection of complex sites is a recently emerging discipline, exemplified by airborne inspection of remote heating networks, or thermographic imaging of city districts from a road vehicle, reported by Stilla and Hoegner [StH08] who used a CityGML model to associate their results.

Autonomous mapping of man-made sites *as-built* is a key competence for these new applications. No valid and up-to-date CAD model may be available from the outset. In any case, it is important to generate and to handle (query, match, recognize, supplement) part-based, metrically and semantically organized geometry models using real measurement data. All inspection results need automatic attachment to such a core description. Constantly the gap between part-structured representations and amorphous point data needs to be bridged.

This paper focuses on industrial installations and procedures for geometric modeling in real time using active sensors, i.e. laser scanners. Components of interest include

1. General structures normally found inside buildings [Koh07] (floors, ceilings, walls, pillars, doors, wall openings, movables)
2. Special support structures mostly found in process plants (props and beams, machine sockets, scaffolds, railings, runways, gates), and, most importantly,
3. Chemical engineering plant machinery: pipe ducts of all shapes, sizes and topologies including flanges and connectors, vessels, pumps, heat exchanges, rotary kilns and other components.

Most examples from the third group are **rotationally symmetric objects** that abound in petrochemical plants. Apparently, rotational symmetry combines the ease of part manufacturing with basic technological advantages like minimum flow resistance and uniform distribution of thermal or pressure load. According to early surveys in solid modeling (Requicha and Voelcker [ReV82]) about 95% of fabricated components can be approximated either by planes, or by rotationally symmetric surfaces like spheres, cones, cylinders, and toroidal surfaces. An immediate goal is to extract the symmetric portions from scanned data and model the underlying geometries compactly, in order to use them as natural landmark features for mapping and also as targets for inspection. Figure 1-1 illustrates a simple example from the experimental waste incineration plant THERESA at the KIT North Campus: a 2D infrared image of the rotary kiln is mapped as a thermal texture onto a cylindrical model of this object fitted to part of the point cloud.

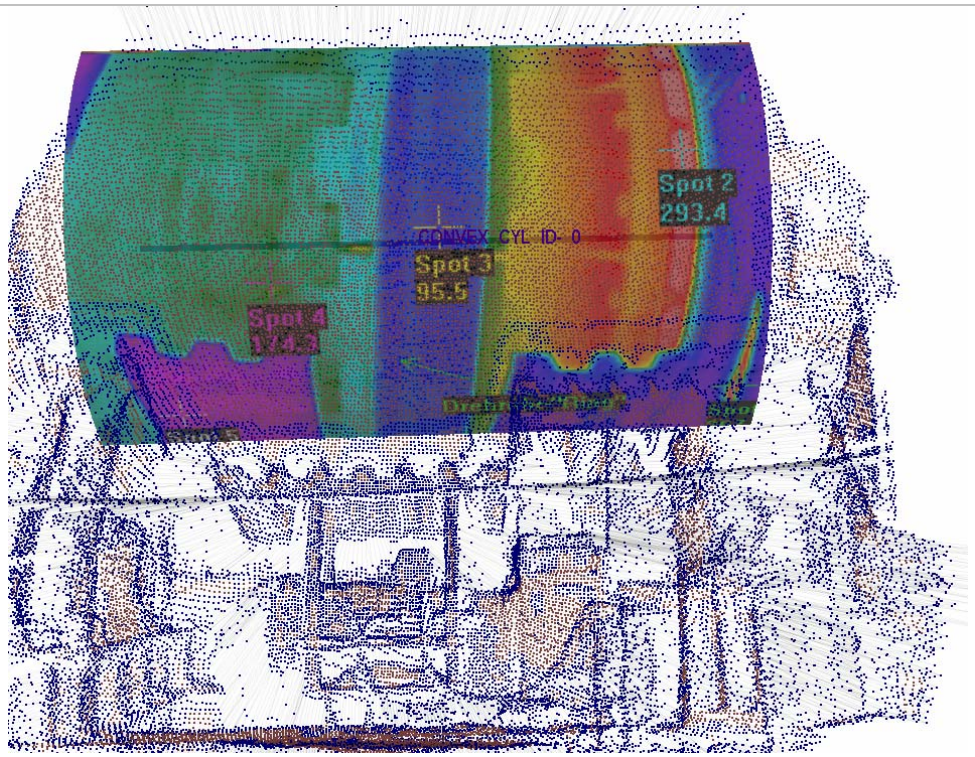

fig. 1-1 A simple example of 3D thermography: a cylinder model of the THERESA rotary kiln estimated from a range image is textured by an infrared image. The range image (captured by a 3D laser scanner in August 2007) and the infrared image (captured during a THERESA campaign in September 2000 from an unknown view point) are not registered.

The mapping and modeling problem stated poses serious scientific and practical challenges. Above all, no restrictive assumptions should be imposed regarding work space illumination, global positioning, prior geometric knowledge, and prior site preparation. We address a casual mapping scenario using affordable laser scanners mounted to a mobile platform or operated in a hand-held or head-mounted fashion. Surveying a spacious installation therefore needs capturing, automatically matching, and aligning many partial views. The objects of interest may be heavily occluded. Despite being rotationally symmetric in space, their visible 2D projection will lack apparent symmetry. The detail geometry may deviate from the gross shape or be obscured by auxiliary objects such as screws, nuts, bolts, valves, flanges, cables, sheathings etc. Prior knowledge from a CAD model may not exist or be readily applicable on-site as to how many rotational objects of which kinds are expected in each view.

When using low-cost scanners, the scene views or pieces of observation cover a limited sector and offer a moderate angular spatial resolution, only. Severe quantization problems arise when estimating, for example, normal vector fields and curvature properties from images, because these operators work on local neighborhoods of points (masks). Obviously, thin objects captured by few points in width are most affected. Images from THERESA in figure 1-2 highlight a few of these problems.

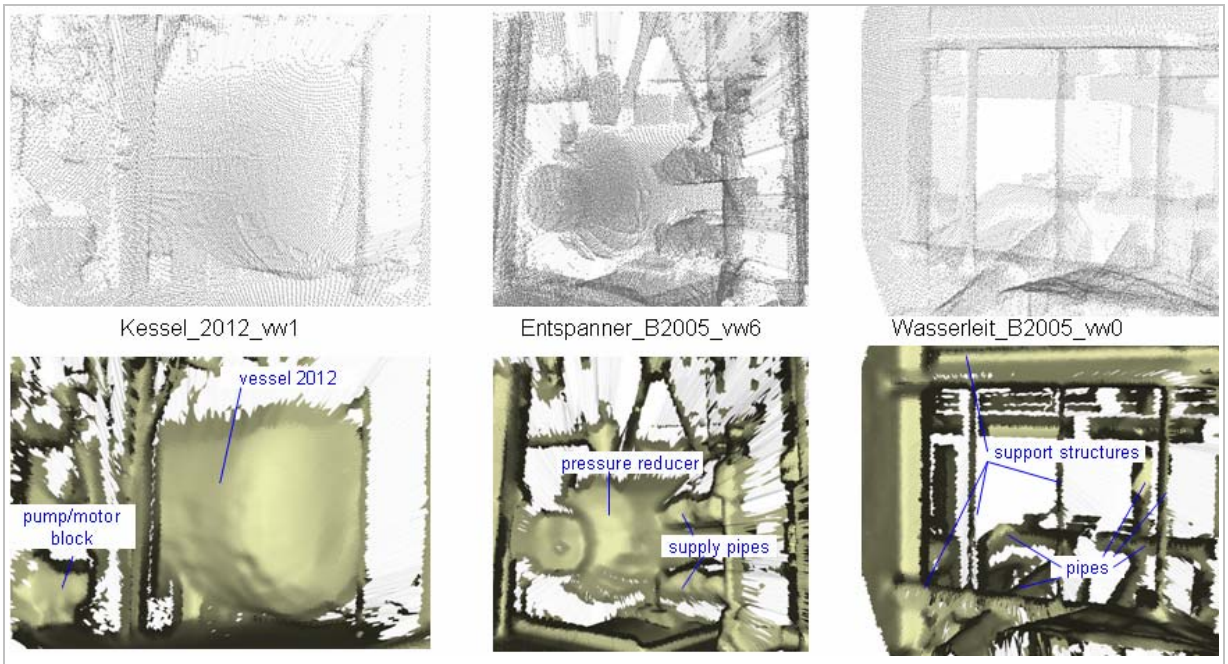

fig. 1-2 Range images from THERESA, top row: 3D points visualized directly, bottom row: shaded rendering obtained from the 'trivial' triangulation by the scanning order (triangles on jump edges and on areas of low point density are suppressed).

No external and global pose reference is available that works reliably in closed spaces and meets the accuracy requirements - especially orientation - of inspection applications. Local and incremental pose estimation, on the other hand, produces a growing uncertainty in the map. Despite moderate accuracy requirements on the final model, only a metrically and topologically *consistent* model is useful: partial features such as pipe ducts or walls are connected and consistently oriented even after closing an exploration loop with a large accumulated error (problem of simultaneous location and mapping, SLAM [TBF06]).

A key goal is to build informative feature maps in real time. Beyond geometry, a coarse functional or hierarchical object classification should be learnt. *Matching features* is essential for pose estimation, for loop closing, and mission related object recognition. Therefore, features are needed *during* exploration, already. Their estimation must keep pace with data capture and, assuming capture times linear in the amount of data recorded, estimation complexity should also be linear or dominated by linear terms. This already precludes many stochastic optimization methods known for labeling, segmentation, and classification problems.

All stages from raw data to parametric SoR descriptions, including component labeling, hypothesis generation, parameter optimization, and hypothesis verification need to be covered. There is a vast amount of work in 3D segmentation and labeling [CRD08] emphasizing on triangulated data, on reverse engineering or medical imaging as application domains and on

methods to discern complex shapes or to optimize their boundaries such as Jagannathan [Jag05], Gelfand and Guibas [GeG04], but rather not object parameter fitting. Conversely, the fitting of cones, cylinders, and other quadratic surfaces using geometric distances or approximations has been thoroughly studied in the CAD domain [AhE03][ARC02][LMM97] [LMM98][MLM01] or in archaeology or cultural heritage preservation [DLP05], for example reconstructing pottery from shards [YaS03][OrW00][Wil04][WiC04]. In this type of work, comparatively little attention is or needs to be paid as to how find efficiently good starting values for optimization.

There is limited recently published work on industrial plant mapping, most notably by Rabbani and Vosselman [Rab06][Rab07]. Their Hough space approach to hypothesis generation is in practice restricted to rather simple objects (cylinders). Demands and constraints in reverse engineering are different: specialized laser scanners [BaF99][Lei09][Rab07] provide panoramic (360° field-of-view) images and a high angular resolution. Only a few views may suffice to cover a large portion of a site and accumulation of pose uncertainties is then uncritical. Comprehensive and distinctive views are easier to register automatically [Rab06], and no real-time processing is required. On the other hand, reverse engineering copes with unorganized point clouds; details of the scanning process are often unavailable. High measurement accuracy of ± 2mm is typically required when planning, for example, a collision-free extension pipe in an industrial plant, whereas accuracy requirements for contactless inspection or collision free navigation are much lower.

We are aware of few real-time robotic applications of SoR modelling with the exception of Taylor [Tay04]; in particular, this applies to the SLAM context. Using segmented surface features in 3D SLAM is now becoming popular but mostly restricted to planar ones [Liu01] [Koh07][HSi07][GFF08]. Again, a crucial problem when dealing with rotational objects and nonlinear optimization is that starting values close to the optimum are needed. With the exception of Rabbani [Rab06] little experience from industrial plants as-built and in use seems to have been published. Quantitative evaluation of the reliability, parameter accuracy, sensitivity, and real-time performance of modelling tools is lacking. Though many methods are known in principle, we feel uncertain about which ones are actually proven and applicable, and hope that this report may help to fill some of the mentioned research gaps.

The remainder of the report is organized as follows. We look more closely at the SLAM mapping problem in chapter 2, and show how to place this piece of work inside an implemented SLAM architecture. Chapter 3 discusses representations for rotationally symmetric objects and introduces basic concepts such as the geometric distance measure used in all algorithms. The approach and system architecture for feature estimation are also explained. With this background, previous work related to different problem aspects is reviewed in more detail in chapter 4. Chapter 5 contains the main technical contributions. Not only a tool box for SoR estimation is developed but a control structure integrating the building blocks to complete and novel algorithms. These algorithms are compared and tested at real range images from the THERESA plant in chapter 6; important properties such as stability with respect to input variations (random point sampling, Gaussian noise, motion noise) and convergence behavior are experimentally evaluated. Chapter 7 presents a short summary with concluding remarks on the lessons learned, and an outlook to forthcoming work.

2 The SLAM framework

The building block described in this report fits into a larger SLAM framework [Koh07]. This chapter briefly summarizes the framework and explains the role and interfaces of this building block.

Simultaneous localization and mapping (SLAM) denotes the problem and the associated techniques to build a geometric map from a stream of local observations where the observer path is unknown or, at best, imprecisely known. Map features carry pose, sometimes also velocity, coordinates referring to a *global* coordinate frame, whereas observations refer to *local* view points. When map features bear on estimated view points, and those in turn on recognized map features, their uncertainties mutually depend and tend to grow. Knowing the amount (magnitude) of uncertainty and acting accordingly is a key point discerning SLAM from related problems like 3D reconstruction.

Ideally, the entire probability density function should be estimated for the observer pose as well as all feature poses in the workspace, conditioned on all available measurement and control data. The probability estimates are iteratively updated, using an observation (sensor) model and a motion model as 'plug-in' components [TBF06]. The sensor model predicts the probability of the current measurement data for a map feature assuming a known view point pose; the motion model gives the probability of reaching a new view point knowing the previous view point and the control data applied to steer the observer. The Bayesian framework with general distributions may become computationally intractable, in particular when practical SLAM requirements and design goals are introduced:

- To work in mostly man-made work spaces (buildings, settlements, plants, mines, networks of infrastructure) where regularity and simplicity of shapes comes at the expense of ambiguity and lacking distinctiveness of single features
- To recover truly spatial geometry and spatial motion paths (3D/6DoF)
- To reliably solve the *unknown data association* problem (which observations correspond to which map features?), using entirely natural ambiguous landmark features[1]
- To perform all relevant SLAM work on-line, including loop closing and map consistency, leaving nothing essential to 'post-processing'
- To provide a detailed map useful for action planning (navigation, object recognition, augmented reality, inspection...), and to maintain a level of representation that supports effective decision making at *any time*.

Under this focus, several existing SLAM approaches are discussed in [Koh07]. Core decisions of a SLAM system involve the principal mapping sensor, the nature of observations, and the map representation. The choices made and the consequences are briefly summarized and discussed.

[1] Assuming no reliable and accurate external pose (orientation!) reference, no unique artificial target landmarks, and no (unbounded!) supply of RFID tags dispersed in the environment for self-localization.

First, *active sensing* is used as the main source of information. It provides the depth information most directly, reliably and accurately in badly illuminated plants or mines, in damaged or inaccessible areas. Despite the high spatial resolution at low cost, and the surge achieved by using scale-invariant features (SIFT [Low04]), stereo correspondence or structure-from-motion relying on passive vision may simply lack sufficient light. In our case, a single 3D laser scanner captures the entire work space. The remission values of the laser scanner corresponding point wise to the depth values provide additional texture information. In process plants, the infrared camera used for inspection will also help to discriminate landmark objects by their levels of temperature. Regarding spatial resolution, however, neither remission images nor infrared images can compete with even the cheapest digital camera. The results reported here have not yet exploited these sources of texture information.

Second, *full range images* or *range views* are processed as the units of observation, like in visual SLAM [DRM07] and in a few laser-based 3D SLAM systems [NLH07]. Images stand a *much* better chance than *scan lines* to recover the observer motion path in six degrees of freedom, and the geometry in three dimensions. More importantly, images contain multiple features that are tightly coupled by geometric relations and relatively unaffected by pose uncertainty. Basically the pose invariance of relational constraints or 'certainty of relations despite uncertainty of positions' [Fre04] is the key to disambiguate landmarks that are individually ambiguous. Another advantage lies in the eigenvector-based motion estimates that jointly map *multiple corresponding features*, the ones forming the static part of the environment: they avoid the *linearization of rotation* from which all Extended Kalman Filter related methods suffer.

On the backside, commercial 2D laser scanners today capture 50-200 scan lines, or roughly one image per second, compared to a video frame rate of about 30 frames per second. The type and speed of observer motion, and also the environment, may change strongly between or during the time intervals when range images are captured.

A **third** major design issue is map representation. In the SLAM literature, location based representations including hierarchical grids, for example octrees or BSP trees, or abstract point-like landmark descriptions dominate. Little is usually said about if and how the map is being accessed while growing, to obtain information for action planning, such as detecting and locating objects to inspect, planning a navigable tour maximizing the gain of new information. The aggregation of information, from raw sensor data to entities of planning, is shown in figure 2-1 as an orthogonal dimension to spatial and temporal map integration.

On the bottom is shown the case where the map is integrated at a low level of point clouds. Information is aggregated when needed, and when the exploration is complete. On top, the map is integrated at a level of surface or object features already. Here, raw data are reduced before the SLAM algorithms act. It is worth noting that any overhead *below* the SLAM level is bounded by the new observations from a fixed time interval. The work at and *above* the SLAM level may deal with the entire map, in the worst case. This overhead to retrieve useful map information as a function of exploration time (or raw data size or explored volume) grows the less steeply the closer the map is to the planning level. It is not just because of the smaller map size as redundancy is detected and eliminated, but mainly because of the lower complexity to search in a map that keeps always organized by the planning criteria to be

searched for. Besides, high level features with distinctive attributes help the SLAM algorithm itself performing feature association correctly, for example by reasoning about visibility, occlusion, and correspondence probability.

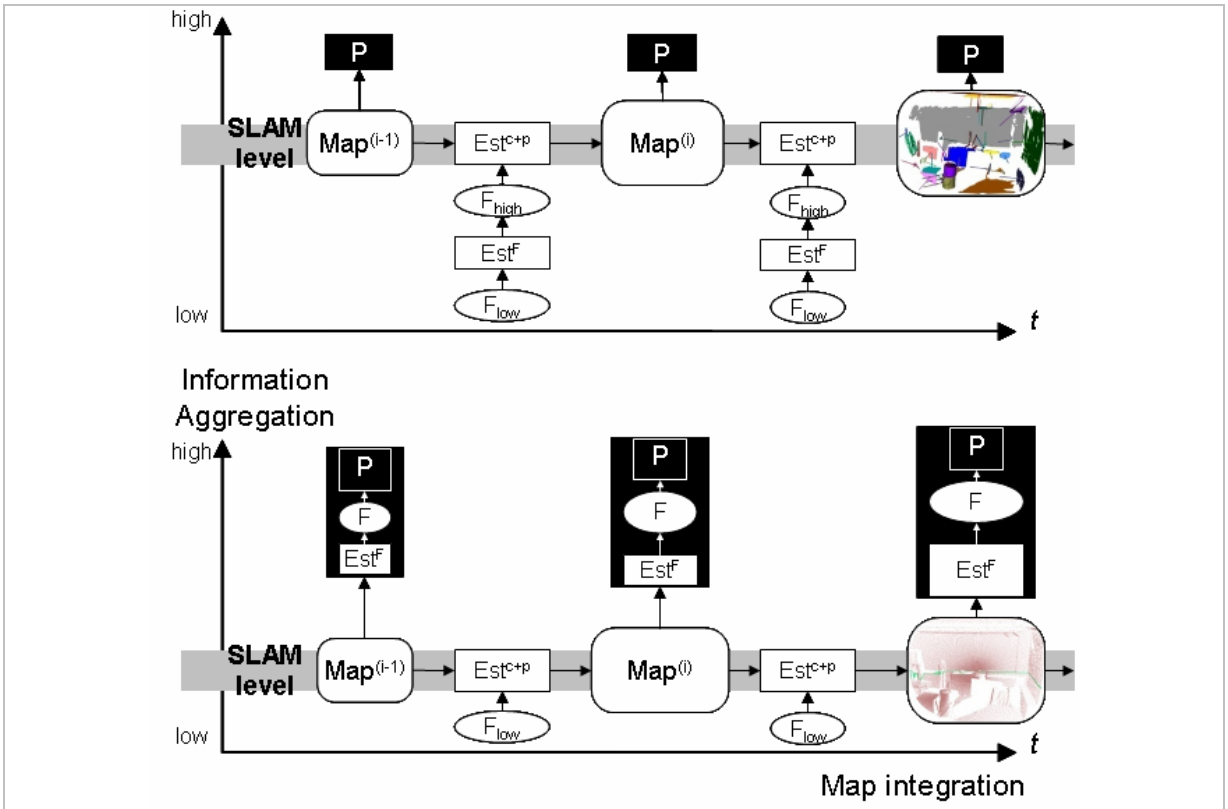

fig. 2-1 Map making may take place at different levels of information aggregation. Ovals symbolize data objects (F_{low} low level, F_{high} high level features), rectangles denote algorithms (P: planning, Est^{c+p}: correspondence and pose estimation (SLAM), Est^{F}: feature estimation).

On the backside, when complex geometric features with pose relations between them are hypothesized and not just range and bearing values read, the assumption of Gaussian sensor models, especially uni-modal density functions will in general become invalid. This is why some of the most efficient SLAM techniques like FastSLAM [TMK04] requiring a Gaussian distribution to perform an EKF landmark update on a particle inside, do not well generalize to 3D features [Sah06].

In this work the high-level mapping approach is chosen, handling sub-maps consisting of oriented surface or object features linked by metrical or topological relations. These are relational attributed feature graphs [ShH82]. Features have nonzero extent, type, and more attributes. From the sensing point of view, a sub-map provides a scene aspect from a single view point or a short view path.

The SLAM framework is divided into two layers: a local layer for estimating correspondence and pose between adjacent feature sub-maps, in general ambiguously, and a global layer for detecting loops, disambiguating poses, and ensuring a consistent global map. A uniform interface between the layers allows plugging in different local algorithms adapted to the specific 3D features and constraints. The interface allows for two kinds of uncertainty: *stochastic uncertainty* and *structural ambiguity* [Koh07]. Structural ambiguity means multiple hypothe-

ses (c_i, $\mathbf{T}_i$) being local optima of some rating function where the choice of the global optimum may randomly depend on input perturbations. In our SLAM case, the c_i denote binary correspondence relations, in general not one-to-one, between the feature sets of two views, and $\mathbf{T}_i$ rigid transformations (3D rotation and translation) applicable to feature poses. The hypotheses are local minima of some cost function or, likewise, maxima of a rating function g that depends on the input features, the correspondence, and the pose transformation. Stochastic uncertainty denotes the variance of pose parameters at the current local optimum given the input variance. It is estimated as a covariance matrix $\mathrm{C}^{\mathbf{Ti}2}$ by covariance propagation, i.e. by linearization of the function assigning the optimum pose parameters $\mathbf{T}_i$ to the current input feature parameters.

Before turning to the local layer where our feature estimation fits into, the function of the global layer is briefly summarized (figure 2-2). Basically, the global layer propagates local uncertainty and compares it to a global consistency criterion called *cycle error* which is used to select the correct hypotheses. Among the multi-hypotheses offered by the local layer, all but a small constant number of most ambiguous alternatives are discarded[3]. Put simply, an alternative hypothesis is ambiguous if a small perturbation of the input is likely to change its ordering by rating among the local optima.

Each combination of alternatives generates a condensed version of the global map in the form of a geometric hash table. The table entries correspond to features and poses dilated by their uncertainty regions, and are used for loop conjecture. Retrieval operations for hash keys from the current view may return intersecting candidate views. Transformations are then estimated - using the local layer - from the current view back to those candidate views, thus forming transformation loops. The resulting cycle error - a rigid transformation that should equal the identity - is compared to the accumulated stochastic uncertainty. If the cycle error is larger, alternative transformations are sought achieving a cycle error that can be explained by stochastic uncertainty. When this condition can be satisfied, a loop is closed and the cycle error evenly distributed among the sub-maps. Alternatives concerning sub-maps on the loop will be deleted and the hash tables updated.

2.1 Feature extraction and pose estimation

The local layer expects for each sub-map a set F of features (parameter representations) and a set R of (binary, ternary etc.) relations among features to estimate correspondence and pose. To quantify the pose uncertainty $C^{\mathbf{T}}$ at the output interface, according input covariance matrices denoted C^f, C^r in figure 2-2 are required and propagated using the derivatives of the pose estimating function [KPW04]. For rating different hypotheses, internal similarity measures are constructed, comparing corresponding features as well as relations between features.

---

[2] Since $\mathbf{T}$ has 7 (3 translation and 4 quaternion rotation) parameters, $C^{\mathbf{Ti}}$ is a 7x7 covariance matrix. Because of the unity constraint on quaternions, only 6 parameters are independent.

[3] The current implementation permits at most 5 alternatives at any moment, leaving $\leq 2^5$ combinations.

The local layer should try to reduce the ambiguity of hypotheses as much as possible. Amendments in figure 2-2 (on the right, in dashed lines) show two ways of doing this. One is to independently track the 6-DoF observer pose using platform sensor data, and assess the likelihood of the vision-generated hypotheses using the associated Mahalanobis distance [Koh07] [KPW04]. Wheel encoders (odometry) and inclinometers could be used for a vehicle-mounted scanner, and a 6D mouse for a hand-held sensor. Another option is to generate constraints of visual consistency and completeness [Koh07] for a specific pose hypothesis, exploiting that surface features are oriented and have finite extent.

The segmentation and feature extraction instance (black box in figure 2-2) is responsible for generating a feature sub-map from organized 3D points, and is the topic of the remainder of this report. As before, the covariance of features and relations must be estimated from specified point covariance, either analytically, using propagation and eigenvector perturbation, or empirically. An example of the latter method will be presented in the experiments chapter 6 in the context of stability analysis.

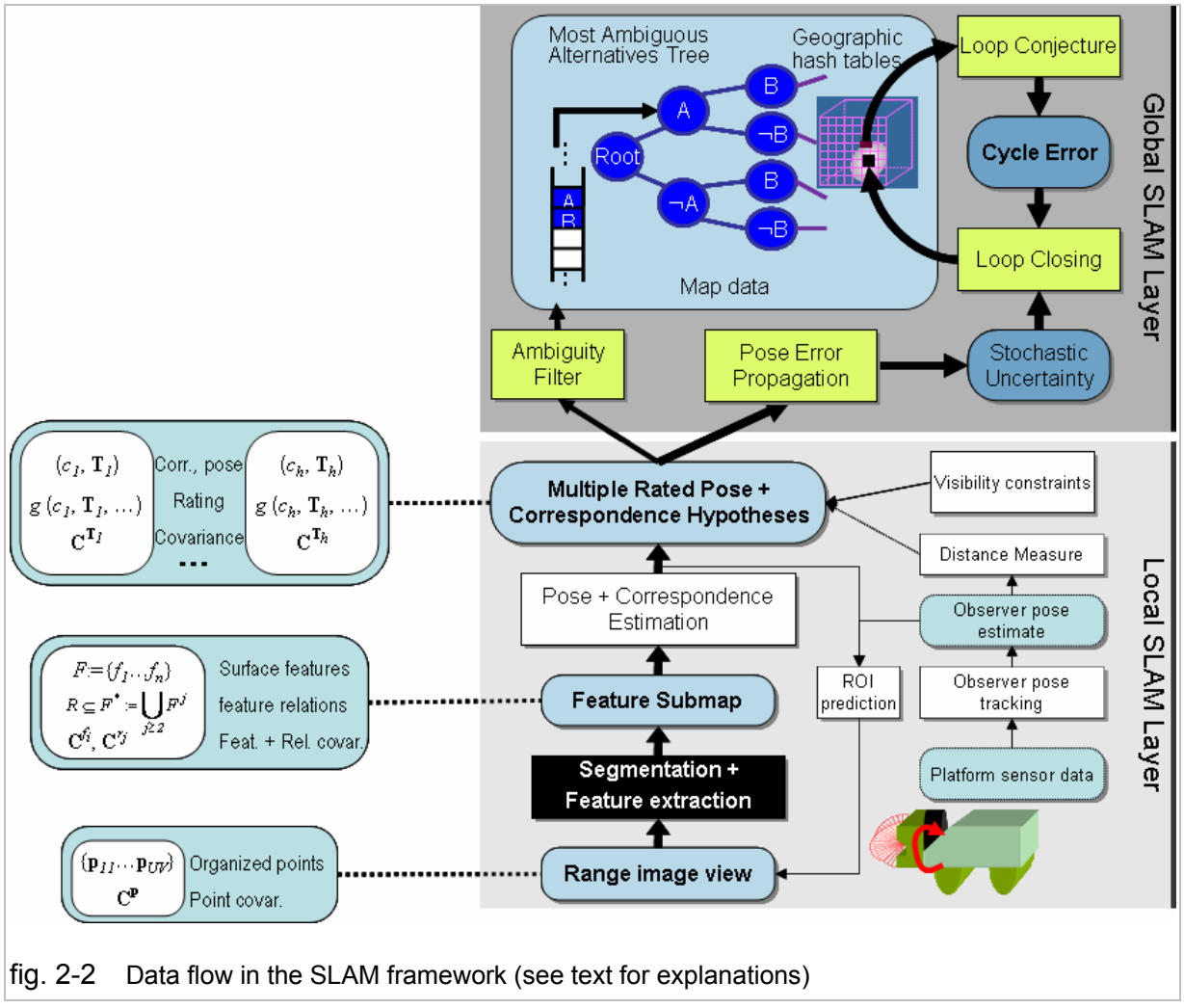

fig. 2-2 Data flow in the SLAM framework (see text for explanations)

Currently, feature maps may contain three classes of surface features: the PATCH (approximating a bounded patch of a planar or low curvature surface), the SOR (representing a visible patch of a rotationally symmetric volume), and the CLUTTER (a bounded patch of a sur-

face with unknown parametric approximation). Surface features share the following attributes:

a) Surface class (PATCH, SOR, or CLUTTER)
b) Outer and inner boundaries of the visible path (closed space polygons, oriented bounding box defined by the principal directions)
c) Center of gravity ($\mathbf{c}$)
d) Mean normal vector ($\mathbf{n}$)
e) Histogram of curvature type signatures (sign combinations, shape classifier)
f) Approximation parameters, approximation error
g) Scalar attributes (visible patch area and extent, degree of occlusion, ...)
h) Weight w (sum of quality values for all points supporting the feature).

No parameters in f) are needed for a PATCH as the centre $\mathbf{c}$ and normal $\mathbf{n}$ completely specify the plane. Surfaces of revolution (SOR) are parameterized by axis and radius function (chapter 3). Complex objects, for example pipe ducts containing different axis curves, are composed of simpler objects by means of grouping relations (section 5.7). CLUTTER surfaces are approximated by lists of triangles; their details (mesh structure, triangle sizes, approximation accuracy) are not covered here. The boundaries b) are not directly used for correspondence and pose estimation but for merging corresponding patches.

2.2 Image capture

For sequential mapping of production plants and similar work spaces, laser scanners with two independently controllable axes are necessary [Koh07]. High-end commercial 3D scanners [BaF99][Lei09] are being used for CAD modelling of large plants and in cultural heritage conservation. For this project, the rotating SICK sensor RoSi developed at Karlsruhe University [SD03] was available: a commercial 2D laser scanner (SICK LMS) rotates about its optical axis and captures a spherical cap with an opening angle δ of 100° (at ¼° resolution) or 180° (at 1° resolution) in front of the scanner (figure 2-3). Power supply and data communication are made of gold slip rings. Owing to the scan line rotation, no orientations are preferred, but objects in the centre receive the highest resolution and at the border of the view field the lowest. The continuous rotation allows fast scanning with low mechanical stress.

For real-time surface extraction the ordering (organization) of points is very helpful. Any point has a scan line index u indicating the current scan line rotation angle $\rho \in [0, 2\pi]$ about the optical axis, and a point index v within the scan line corresponding to the laser beam deflection angle $\alpha \in [-\delta/2, \delta/2]$. The mapping of 2D parameter coordinates to 3D points $(u, v) \rightarrow (x, y, z)$ is basically a Spherical-to-Cartesian coordinate transformation [SD03]. With a fixed view point, the scan lines for $\rho = 0$ and $\rho = \pi$ match in reverse order; each half resolution of RoSi yields a full range image. Therefore, the parameter space topology is a twisted cylinder mantle (Möbius band). In addition, all scan line center points (α=0) are neighbors; point density tends to infinity in the optical center.

Using an opening angle of 100° and an angular resolution of ¼°, each scan line contains 401 points. The sensor throughput amounts to roughly 200 scan lines per second. Operating at 0.5…2 rotations per second yields an angular resolution of 0.9…3.6 degrees and collects

200…50 scan lines per image. In the estimation examples of this report, the raw images were interpolated into angular grids of uniform angles (chapter 6).

The point uncertainty (covariance) depends on the distance measurement noise (in [mm] at 1σ according to the manufacturer), the rotation measurement noise (including the residual mechanical imbalance of the rotation unit), and the observer motion during image capture.

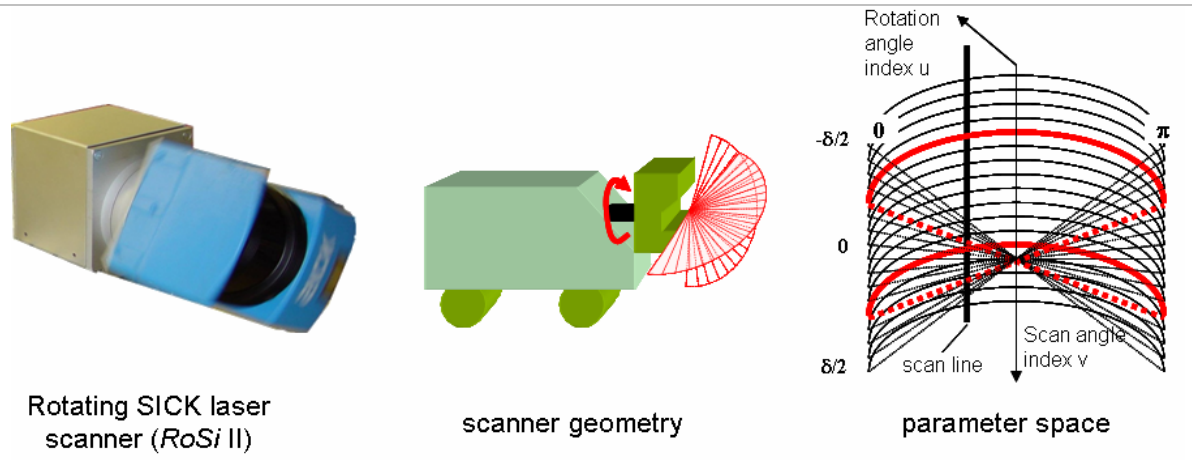

fig. 2-3 Rotating scanner principle

3 Representation and estimation design

3.1 Object representation

The choice of surface or volume representation determines the class of objects that can be faithfully modeled. It influences the numerical accuracy, stability, and the computational complexity of modeling as well as the benefits of using such a representation, for example for object recognition or texture mapping (thermal or olfactory inspection). Some common representations will be briefly summarized and our particular choices explained. For an in-depth treatment of representation issues, the reader is referred to Ahn et al. [ARC02], to the survey by Petitjean [Pet02], and to the extensive CAD literature on solid modeling. Early uses of CAD representations in computer vision are found in [FlJ91].

Industrial geometries, including rotationally symmetric surfaces can be defined by ***implicit functions***: the points $\mathbf{x} \in \Re^3$ lying on the surface satisfy the equation $F(\mathbf{x}, \mathbf{b}) = 0$, i.e. they form the zero set of a scalar function F. Often, the function F is a polynomial in $\mathbf{x}$ with coefficient vector $\mathbf{b}$, hence F is linear in the coefficients. Important cases include quadrics (2nd degree polynomials, representing e.g. hyperbolic and parabolic surfaces, spheres, cylinders and cones), quartics (4th degree polynomials representing tori), or superquadrics (higher degree polynomials in the coordinate magnitudes, describing for example different rounded box shapes). Properties describing the object shape and size as well as its pose in a specific coordinate system are mixed in the coefficient values [ARC02].

When points $\mathbf{x}_i$ are samples from an implicit surface but corrupted by some noise, the residual $F(\mathbf{x}_i, \mathbf{b})$ called the *algebraic distance* directly indicates an error value. Fitting a function to the data, i.e. determine the unknown coefficients so as to minimize the (mean squared) residuals, admits a solution in closed form, since the algebraic distance is linear in $\mathbf{b}$. But algebraic fitting methods are known to suffer from numerical stability problems [ARC02]. The fitting results depend on the specific coordinate system. It is difficult to robustly determine the right *type* (order) of the function F, deciding for example if data are more faithfully approximated by a toroidal cap or by a spherical cap.

Another widely used *surface* representation is by ***explicit parametric functions*** $\mathbf{x} = F(\mathbf{u})$. Here, the vector $\mathbf{u}$ denotes (u, v)-coordinates in a 2D parameter space like polar or cylindrical coordinates uniquely characterizing the 3D points. For example, each point $\mathbf{x}_i$ in a range image captured from a fixed point by a rotating 3D scanner is given by $\mathbf{x}_i = F(u, v)$, where v encodes the deflection angle within a scan line, and u the rotation angle of the scan line. NURBS (non-uniform rational B-splines) are an important class of surfaces described by two parameters u and v in [0, 1]. Problems of fitting NURBS functions to data lie in the estimation of knots and control points. Explicit functions, in general, are not well suited for manipulating, in particular *matching* and *comparing* objects, because the representations do not uniquely characterize an object.

As to manipulating, ***volume descriptions*** by parameterized ***primitives*** have clear advantages. Simple examples include box, sphere, cone/cylinder, and torus as primitives. The specific parameters for each primitive are separated with respect to shape, size, and pose. Contrary to the algebraic distance used with implicit functions, the *geometric distance* is associated with primitives fitted to 3D points: their shortest Euclidean distances orthogonal to the surface are calculated. This makes the optimization pose-invariant but the geometric distance is a nonlinear function of the unknown parameters. Robust fitting methods using geometric distances are presented by Ahn et al. [ARC02]. To overcome numerical problems arising from particular terms in the distance function, faithful approximations of the geometric distance have been proposed in the context of rotation primitives [LMM98].

Real parts, in general, can only be expressed as combinations of several primitives by means of Boolean operations (union, intersection, difference). Objects are described by CSG trees; their nodes represent primitive or composite volumes and the edges operators acting on them. This compact representation lends itself for handling objects, including comparison and matching. Methods to fit an entire CSG tree, such as a flanged pipe or a T-junction, to a single point set have recently been reported [Rab06][Rab07]. Such a procedure requires the building principle to be explicitly specified for each kind of object. This prior knowledge may not be readily available on-site when exploring, say, a legacy plant of non-standardized design, considering alone the many pipe bends of unknown shape and bending radii. CSG primitives are rarely used to model natural (biological) tubular systems like blood vessels, so their modeling power and generality seems to be a limiting factor. Complex-shaped technical installations, e.g. flanges arranged on a pipe, captured with inadequate spatial resolution may however appear similarly 'natural'.

Surfaces-of-revolution (**SoR**) and related subclasses of generalized cylinders (GC) are special primitives focused on rotational symmetry. SoR are defined by an *axis* curve of rota-

tion $\mathbf{a}(s)$, $\mathbf{a}$: $\Re \rightarrow \Re^3$, and a *planar profile curve* or *generatrix* $g(s, r) = 0$. A SoR is a union of *circular cross sections*, each cross section being the locus of points rotated about a fixed axis point $\mathbf{a}(s)$ at distance r such that (s, r) lies on the profile curve ($g(s, r) = 0$) and the rotation plane is orthogonal to the axis at $\mathbf{a}(s)$. Usually, the axis is assumed to be a straight line. Then the profile curve is often written as an explicit scalar *radius function* $r(s)$ in the (s, r)-plane. *Continuous* space curves $\mathbf{a}(s)$ allow for describing curved tubular structures.

Cylinder, cone, and sphere are simple special cases; spheres with their axes passing through the center in arbitrary direction are mostly attached in practice to an oriented object like a cylinder with spherical closure. Toroidal surfaces most often occur as open torus segments, such as 90° pipe bends. There are two ways to describe torus segments in a 'SoR-like' fashion:

- Consistently with the torus definition, by the torus main symmetry line, normal to the torus plane and passing through its center. The profile curve is a circle, and the radius function reads $r(s) = R_T \pm \sqrt{r_T^2 - s^2} \quad (-r_T \le s \le r_T)$, R_T denoting the outer and r_T the inner torus radius.
- As a tube with a circular or piecewise linear *arc* as the axis curve, and a *constant* radius function $r(t)=r$. Analogously to the medial axis of a straight tube, the *medial axis* of the torus ring is then seen as its axis.

Straight-axis SoR descriptions have the advantage of representing the surface uniquely, when both the axis and the radius function are specified uniquely, using a minimal set of independent parameters. Therefore, SoR admit simple and effective similarity functions to compare and match shapes. As to their power of representation, SoR form a good compromise between generality and compactness, approximating many man-made objects in different domains, like bottles, drinking vessels, vases, bells, and tools easily.

Complex components in industrial plants are composed of pipe and vessel primitives and linked by relational constraints. From a CSG point of view these are mainly unions, but more specific information is gained by classifying the kind of geometric relation between the axes. Individual relations between elements of type adjacent, connected, coaxial, coplanar, branching and others are considered. Equivalence relations are thereby imposed on a set of primitives, leading to classes the elements of which belong to a common duct.

Reliable estimation of SoR structures from range images, encompassing segmentation and fitting, remains however difficult. Early work on recognizing SoR or GC from perspective camera images [SaB93][ZeN96] exploits and requires strong symmetries in the 2D appearance [CBP05], i.e. limited occlusion. More recent applications of SoR using 3D data are found in medical imaging (e.g. artery reconstruction [WoC08][BRS03]) and in cultural heritage conservation, e.g. modeling architectural features like balustrade pillars [DLP05] or reconstructing pottery from shards [YaS03][Wil04][WiC04]. Especially the latter application exemplifies difficulties also found in industrial sites: estimating independent direction and radius function parameters from arbitrarily shaped or heavily occluded parts that cover only a *small arc of curvature*. Reconstructing axes of rotational symmetry from partial 3D data has been studied with a theoretical focus notably by Pottmann's research group. By applying algebraic line geometry, they recognize even broader classes of rotationally symmetric surfaces, e.g. those generated by uniform equiform motion [HOP05][OPW05].

Summarizing, the surface-of-revolution with a straight line axis segment and an arbitrary scalar radius function will be used to represent the basic object primitives. *Circular arc* axes may be considered as a special case, or, in practice, are approximated by an (open) 3D polygon. The straight axis is simply a 3D line $L(\mathbf{s}, \mathbf{a})$ passing through some point $\mathbf{s}$ in the (unit) direction $\mathbf{a}$. The radius function $r(s)$ is defined over a bounded interval r: $[s_{min}, s_{max}] \rightarrow \Re$, where the scalar variable s denotes position along the axis direction. A radius function can be drawn as a planar curve, and, moreover, can be approximated arbitrarily closely by a piecewise linear function (a polygon). The SoR is thereby reduced to a sequence of truncated cones contiguously lined up on common axis and with continuous radius function.

3.1.1 Normal ray property

In this sub-section, an important property for the estimation of SoR parameters is discussed; it is best understood using smooth (differentiable) space curves [ZeN96]: let a rotationally symmetric surface be characterized by a *smooth axis curve* $\mathbf{a}(s)$: $[0,1] \rightarrow \Re^3$ and a *smooth radius function* $r(s)$: $[0,1] \rightarrow \Re$ generating the cross sections. The tangent direction at $\mathbf{a}(s)$ is the derivative of the axis $\mathbf{t}(s) := \dot{\mathbf{a}}(s)$. If $\mathbf{t}(s)$ itself is differentiable, the derivative $\mathbf{n}(s) = \dot{\mathbf{t}}(s) / \|\dot{\mathbf{t}}(s)\|$ equals the unit normal direction [NuM88] and lies in the cross section plane. Every point on the cross section circumference is characterized by an angle of rotation $\theta \in [0, 2\pi]$ included with the normal direction $\mathbf{n}(s)$. The entire surface S has an explicit parametric representation by arc position s and rotation angle θ:

$$S(\theta, s) = \mathbf{a}(s) + r(s) \cdot \left[\cos(\theta) \cdot \mathbf{n}(s) + \sin(\theta) \cdot \mathbf{b}(s) \right] \quad (\mathbf{b}(s) = \mathbf{t}(s) \times \mathbf{n}(s) \quad binormal) \tag{3.1}$$

The general representation (3.1) allows 'invalid' (non-orientable) surfaces. For example, self-intersections may occur if the axis curve has sharp bends with a curvature radius smaller than the SoR radius. The definition assuming smooth curves may be extended to piecewise linear radius and axis functions. In [KoF99] 3D polygons were used as curves by defining suitable curvature and torsion functions. To obtain a continuous curve, at every break point s a small interval $[s, s+\Delta s]$ is inserted to replace the normal discontinuity by a constant-velocity rotation $\mathbf{n}(s)$ while keeping the radius function and the bi-normal (orthogonal to both adjacent edges) constant.

Assuming a straight-axis SoR in its *standard* parameterization, the axis will be aligned with the z-direction: $\mathbf{a}(s) = (0, 0, s)^T$, and the parameterized surface (3.1) takes the form

$$S(\theta, s) = \begin{pmatrix} r(s)cos(\theta) & r(s)sin(\theta) & s \end{pmatrix}^T \tag{3.2}$$

For a fixed scalar value s_0, $S(\cdot, s_0)$ describes a *circular cross section* with radius $r(s_0)$ and center point $(0, 0, s_0)^T$. For a fixed rotation angle θ, $S(\theta, \cdot)$ is the equation of the generating *curve* $r(s)$ rotated so as to lie in the *plane* $E_\theta := \{(x,y,z)^T: \; -x \cdot sin\,\theta + y \cdot cos\,\theta = 0\}$. At any surface point, the normal vector is perpendicular to the two tangent directions defined by the partial derivatives of the surface (3.2) with respect to both parameters, s and θ:

$$\partial S(\theta,s)\Big/\partial s = \left(\dot{r}(s)\cdot\cos\theta \quad \dot{r}(s)\cdot\sin\theta \quad 1\right)^T$$
$$\partial S(\theta,s)/\partial\theta = \left(-r(s)\cdot\sin\theta \quad r(s)\cdot\cos\theta \quad 0\right)^T \tag{3.3}$$

The (unit) normal direction is therefore obtained as the vector product

$$\mathbf{n}(\theta,s) = (\partial S(\theta,s)/\partial\theta)\times(\partial S(\theta,s)/\partial s) = \frac{1}{\sqrt{1+\dot{r}^2(s)}}\cdot\left(-\cos\theta \quad -\sin\theta \quad \dot{r}(s)\right)^T \tag{3.4}$$

Casting a ***normal ray***, a line $L(\mathbf{p}, \mathbf{n})$:= $\{\mathbf{p}+t\cdot\mathbf{n} \mid t\in\Re\}$ passing through surface point $\mathbf{p} = S(\theta, s)$ in the normal direction $\mathbf{n}(\theta, s)$, and substituting (3.2) and (3.4) shows that the normal ray intersects the *z* (SoR) axis for $t = r(s)\sqrt{1+\dot{r}^2(s)}$, yielding the intersection point

$$\mathbf{p}_L = \left(0 \quad 0 \quad s + r(s)\cdot\dot{r}(s)\right)^T$$

The intersection exists, of course, for any parameterization since rotation and translation transform all points in the same way. This basic fact is referred to as the *normal ray property* and will be useful for detecting and seeding the parameters of SoR with straight axis:

Every *normal ray* $L(\mathbf{p}, \mathbf{n})$ to a smooth surface of revolution (SoR) with straight axis and circular cross sections intersects the axis line at some point $\mathbf{a}(s)$. (3.5)

The normal ray property (3.5) characterizes *smooth* SoR with straight axis in the sense that all cross sections are circular *if and only if* all normal rays intersect the axis. Property (3.5) does not hold for curved or piecewise linear axes. On noisy digitized surface data, the normal rays give an error criterion for testing a SoR hypothesis: the (root of) mean squared distances of all normal rays from the known axis. In case of non-smooth surface data, a small distance value is necessary, but no longer sufficient to claim a truly rotationally symmetric surface. Figure 3-1 illustrates an *asymmetric* non-circular cross section where sectors with different radii are separated by jump edges. Yet the ray distance will be presumably small: the majority of normal rays are well-centered. The few points on the jump edge violating the normal ray property contribute little to the distance (points on jump edges may be 'smoothed away' by normal vector estimation, i.e. no discontinuity may be recognized).

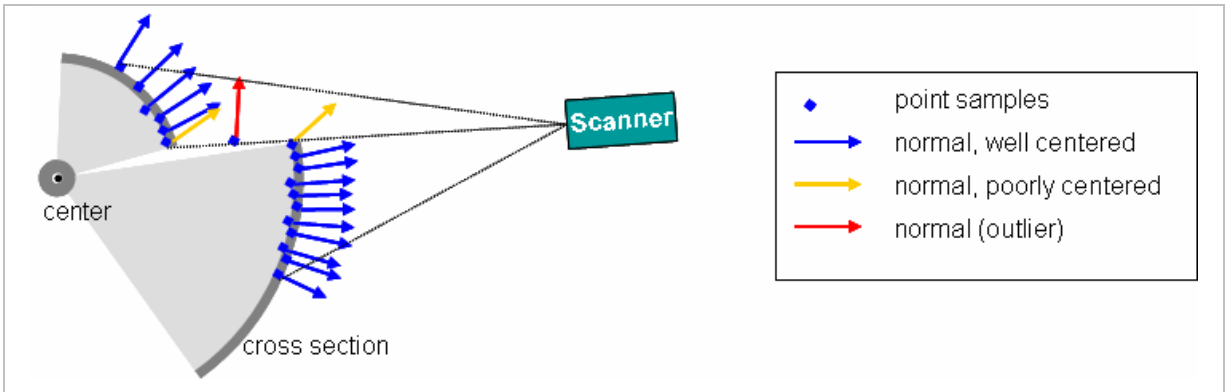

fig. 3-1 Example of a scanned cross section with a small normal ray error which is not a valid SoR cross section.

Additionally testing the degree of *radial symmetry* is therefore needed on sample data. A point property parameterized by s (cross section) and θ (radial angle or meridian), such as the axis distances $r(\theta, s)$ or the Gaussian curvatures $K(\theta, s)$ may be used to check that their distributions, for example histograms binned with respect to s and θ, do not depend on the angle θ. The cross-correlation between the meridians can be applied.

3.1.2 Distance measure and closest points

For distance calculations, the following notations and basic geometric facts will be used throughout (see figure 3-2). The rotation axis of a straight SoR is specified as a line $L(\mathbf{s}, \mathbf{d})$ with direction vector $\mathbf{d}$ which passes through a reference point $\mathbf{s}$. For a point $\mathbf{p}$, $\Delta\mathbf{p} := \mathbf{p}–\mathbf{s}$ denotes the difference vector with respect to $\mathbf{s}$, and the unit vector $\mathbf{a} := \mathbf{d}/\|\mathbf{d}\|$ the axis direction. The projection length $s(\mathbf{p}, L)$ of point $\mathbf{p}$ onto L, the orthogonal distance $d(\mathbf{p}, L)$ between $\mathbf{p}$ and L, and the axis projection vector $\mathbf{p}_\perp(\mathbf{p}, L)$ onto L are defined by

$$\begin{aligned} s(\mathbf{p}, L) &= \Delta\mathbf{p}^T \cdot \mathbf{a} = \|\Delta\mathbf{p}\| \cdot \cos\angle(\Delta\mathbf{p}, \mathbf{a}) \\ \mathbf{p}_\perp(\mathbf{p}, L) &= \Delta\mathbf{p} - s(\mathbf{p}, L) \cdot \mathbf{a} = \left(\mathbf{I}_3 - \mathbf{a} \cdot \mathbf{a}^T\right)\Delta\mathbf{p} \\ d(\mathbf{p}, L) &= \|\mathbf{p}_\perp(\mathbf{p}, L)\| = \sqrt{\Delta\mathbf{p}^T \cdot \Delta\mathbf{p} - s^2(\mathbf{p}, L)} = \|\Delta\mathbf{p}\| \cdot \sin\angle(\Delta\mathbf{p}, \mathbf{a}) \end{aligned} \tag{3.6}$$

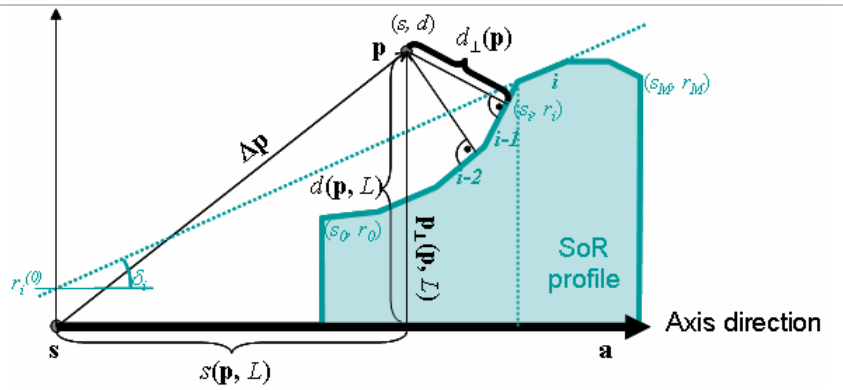

fig. 3-2 Profile plane section through a surface of revolution (SoR); identifiers are used for distance calculations and explained in the text.

Often, the SoR radius function is piecewise linearly approximated by a sequence of 2D break points (s_i, r_i) with projection values $s_0 \ldots < s_i < \ldots s_M$ $(0<i<M)$ along the axis and radii $r(s)$. The linear radius segment between two break points is defined by an intercept value $r^{(0)}$ at $s=0$ and a slope angle δ. The radius function is then written as a parameter sequence

$$\left\{ \left(r_i^{(0)}, \delta_i\right) \middle| \, 0 \le i < M, \quad \delta_i = \tan^{-1}\frac{r_{i+1} - r_i}{s_{i+1} - s_i}, \quad r_i^{(0)} = r_i - s_i \tan\delta_i \right\} \tag{3.7}$$

A point $\mathbf{p}$ represented by $(s, d) = (s(\mathbf{p}, L), d(\mathbf{p}, L))$ has orthogonal distance $d_{i\perp}(\mathbf{p})$ from the i-th *unbounded* cone surface which is given by

$$d_{i\perp}(\mathbf{p}) = \left| \left(r_i^{(0)} - d\right) cos\,\delta_i + s\, sin\,\delta_i \right| . \tag{3.8}$$

SoR in this work mean *bounded* surfaces, unions of *truncated* cones, and bounds are relevant in measuring distances. The distance of point $\mathbf{p}$ from the surface $d_\perp(\mathbf{p})$ equals the shortest distance among cones onto which $\mathbf{p}$ projects (let $M_\perp(\mathbf{p}) := \left\{ i \,|\, s_i < s + d_{i\perp}(\mathbf{p}) \cdot sin\delta_i < s_{i+1} \right\}$ denote the index set), and equals the shortest distance $d_i(\mathbf{p}) := \sqrt{(d-r_i)^2 + (s-s_i)^2}$ from a break point, if $\mathbf{p}$ projects onto none of them:

$$d_\perp(\mathbf{p}) = \begin{cases} \min\left(\min\limits_{i \in M_\perp(\mathbf{p})} d_{i\perp}(\mathbf{p}), \min\limits_i d_i(\mathbf{p}) \right) & if\ M_\perp(\mathbf{p}) \neq \varnothing \\ \min\limits_i d_i(\mathbf{p}) & else \end{cases} \tag{3.9}$$

Besides positive distances, signed quantities are also used. A linear segment within the SoR profile divides the profile plane into two half planes E_+ ('in normal direction'), E_- ('opposite to normal direction'), using the projected surface normals. By assigning each point $\mathbf{p}$ having coordinates (s, d) in the profile plane to a unique i-th line segment ($i=0 \Leftrightarrow s<s_1$, $0<i<M-1 \Leftrightarrow s_i \leq s < s_{i+1}$, $i=M-1 \Leftrightarrow s \geq s_{M-1}$), a point has a side $sd(\mathbf{p}) \in \{1,0,-1\}$. Also, considering the surface normals, the entire SoR surface S gets a sign $sgn(S) \in \{1,-1\}$ defined to be 1 for convex cross section circles and -1 for concave cross sections (hollow surface).

When distances to a fine-grained radius function are frequently calculated, determining the closest break point for a given query point $\mathbf{p}$ is faster than finding all segments onto which $\mathbf{p}$ projects. Therefore, the closest distance $d_\perp(\mathbf{p})$ is simplified by an approximation which is justified if the line segments are of fairly uniform length: the closest *break point* is located and its right and/or left neighbor segment(s) are inspected, only, if a projection exists:

$$\begin{aligned} &\widetilde{M}_\perp(\mathbf{p}) := M_\perp(\mathbf{p}) \cap \{j-1, j\} \quad where \quad j = \arg\min_i d_i(\mathbf{p}) \\ &\widetilde{d}_\perp(\mathbf{p}) = \begin{cases} \min\limits_{i \in \widetilde{M}_\perp(\mathbf{p})} d_{i\perp}(\mathbf{p}) & if\ \widetilde{M}_\perp(\mathbf{p}) \neq \varnothing \quad (projection\ \ onto\ segment) \\ d_j(\mathbf{p}) & else \quad\quad\quad\quad\quad\quad (break\ \ point) \end{cases} \end{aligned} \tag{3.10}$$

Accordingly the SoR surface point $\mathbf{p}'$ on closest to $\mathbf{p}$, and the normal vector $\mathbf{n}'$ at $\mathbf{p}'$ are found by cases, see fig. 3-2. In the 'projection' case in (3.10), $\mathbf{p}'$ is related to $\mathbf{p}$ by travelling a distance $d_\perp(\mathbf{p})$ against or in the normal direction $\mathbf{n}'$, depending on the side $sd(\mathbf{p})$. The surface normal $\mathbf{n}'$ lies in the profile plane and includes an angle $\pm(\delta_j + \pi/2)$ with the axis direction $\mathbf{a}$. In the 'break point' case, the point $\mathbf{p}'$ is found directly from the parameters (s_j, r_j) and the axis projection $\mathbf{p}_\perp(\mathbf{p}, L)$ (3.6). The corresponding normal vector $\mathbf{n}'$ as such is undefined (the surface S is not differentiable at break points), but is supplemented from the normalized connection vector $\mathbf{p}'$-$\mathbf{p}$ and consistently with the surface sign, $sgn(S)$:

$$\mathbf{p}'(\mathbf{p}) = \begin{cases} \mathbf{p} - sd(\mathbf{p}) \cdot \widetilde{d}_\perp(\mathbf{p}) \cdot \mathbf{n}', \quad \mathbf{n}' := \mathbf{R}\left(\dfrac{\mathbf{a} \times \Delta\mathbf{p}}{\|\mathbf{a} \times \Delta\mathbf{p}\|}, sgn(S) \cdot (\delta_j + \pi/2) \right) \cdot \mathbf{a} & if\ projection\ to\ segment\ j \\ \mathbf{s} + s_j \mathbf{a} + \dfrac{r_j}{\|\mathbf{p}_\perp(\mathbf{p}, L)\|} \mathbf{p}_\perp(\mathbf{p}, L), \quad \mathbf{n}' := sgn(S) \cdot sd(\mathbf{p}) \cdot \dfrac{\mathbf{p}' - \mathbf{p}}{\|\mathbf{p}' - \mathbf{p}\|} & if\ break\ point\ (s_j, r_j) \end{cases} \tag{3.11}$$

$\mathbf{R}(\cdot, \cdot)$ in (3.11) denotes a rotation matrix about the axis given as the first argument and the rotation angle as second argument; the rotation axis is orthogonal to the profile plane.

3.2 Estimation Design

The algorithms to be developed implement and refine the feature extraction instance in the SLAM data flow diagram, shown in black in figure 2-2. At the coarsest level, the work is divided into four processing stages forming a pipeline (figure 3-3)

- Point feature mapping and connected component labeling
- Hypothesis generation
- Hypothesis optimization via EM (expectation-maximization)
- Hypothesis evaluation, verification, and relation estimation.

The last estimation stage generates information that may cause alternative hypotheses to be generated, explaining the control loop back in figure 3-3. The processing instances are coupled via the following principal data structures illustrated in fig. 3-3:

- Point models at different levels of integration: from input image (organized 3D points) to oriented surface points carrying additional features of curvature, shape, edge strength, and others.
- Connected components (CC): regions of points that are connected by the scanning order and are homogeneous with respect to some characterizing property, for example curvature or shape type.
- SoR hypotheses with object parameters including bounds and uncertainty (variance, covariance) estimates of parameters. A hypothesis refers to a single element, a straight-axis SoR or its special cases (cone, cylinder). The support set initialized from or referring to a CC is part of the current hypothesis, and so is a criterion of quality and weight for ranking the hypothesis. The quality attribute will gradually change its role, turning from a degree of promise during hypotheses generation to an error-based likelihood during hypothesis optimization and finally into a degree of confidence at hypothesis evaluation.
- SoR group hypotheses: they are ordered sets of SoR elements linked by some relation. The individual elements may be connected merely topologically when their point sets are adjacent in the input image, or be linked algorithmically, for example being generated by the same method invocation, or be physically or functionally connected (coaxial, coplanar, branching, or parallel) pipe sections. A SoR group has a unique group identifier, the component ID's, the relation type, and relation attributes, e.g. the description of a common line or plane, included angle or distance values.

The application, i.e. SLAM for inspection in industrial installations, demands part descriptions to be estimated timely, reliably, and accurately from partial views. The step responsible for building the final models is the optimization stage; the others can be seen as prerequisites and necessities derived from it. We do not know beforehand how many interesting objects appear in an image, of which kind and where. Finding and modeling objects therefore is a chicken-and-egg problem: knowing the points sampled from specific target objects enables to estimate the model parameters best explaining the data (fitting) and, conversely, knowing the parameters allows assigning the points that best support the model. Expectation-Maximization (EM) is a well-known paradigm integrating and alternating both tasks: the expectation (E-)step lets several model hypotheses compete for data points (maximum likeli-

hood observations), and the maximization (M-)step adjusts the model parameters to the previously assigned data, maximizing the model likelihood or minimizing the fitting error. Selecting an EM algorithm for SoR hypothesis optimization was inspired by work of Liu et al [Liu01] who used it in a SLAM context to find planes (unbounded, infinite ones) in an indoor environment. Some pre- and post-processing is needed as well, e.g. to determine the number of planes. Unlike planes, no closed solution is available for fitting of cones/SoR using the geometric distance, only *iterative* optimization. EM in total converges to a *local* optimum of likelihood in the space spanned by the data partitions and the model parameters.

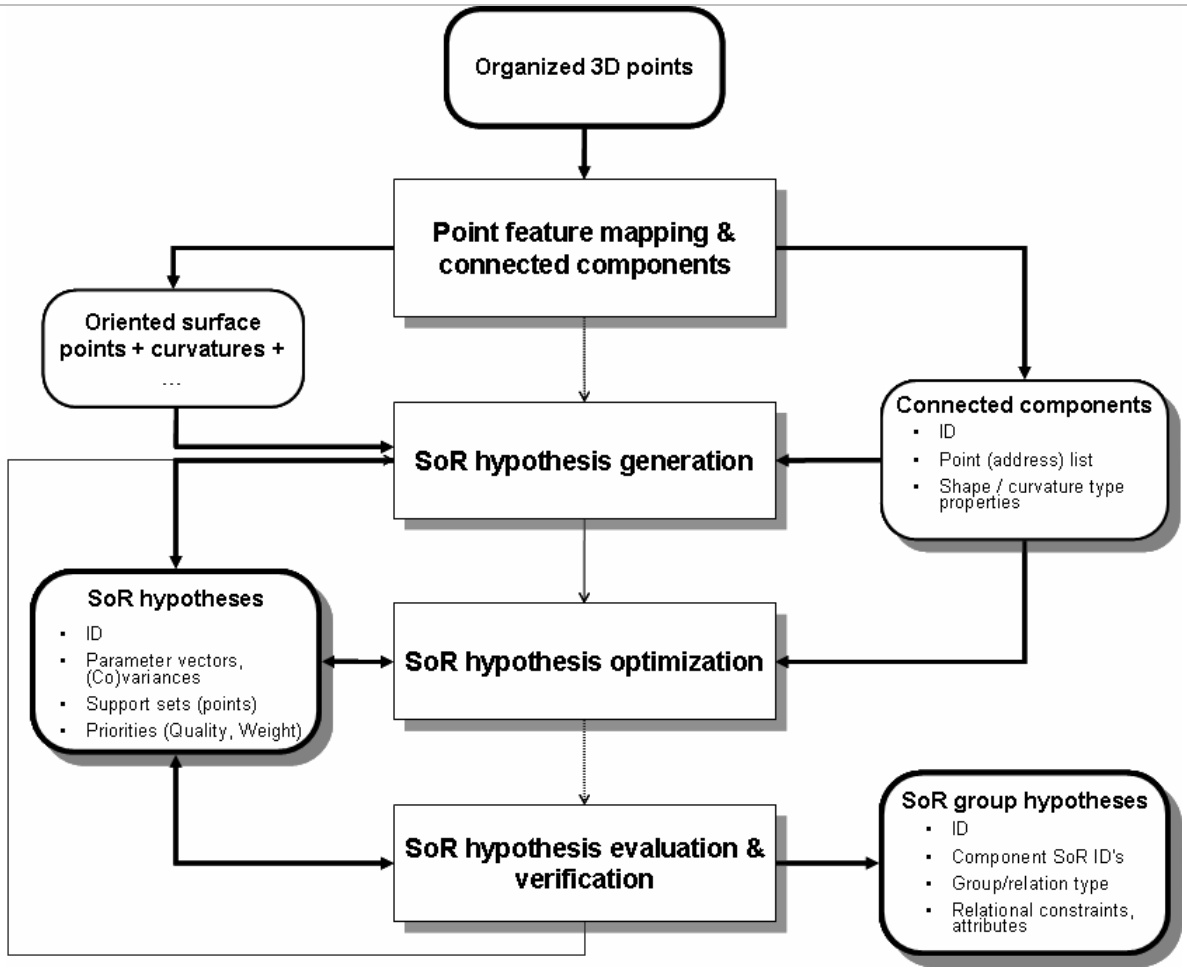

fig. 3-3 Processing stages and data flow for SoR detection and modelling

Local and *iterative* are the keywords calling for a dedicated step to estimate the number of object hypotheses, and to generate excellent starting values that will gear a purely local optimizer (EM) with high confidence to near-optimum parameter values (fig. 3-3). It follows that the sought methods must not themselves *depend* on good starting values; preferably *direct*, i.e. non-iterative, methods yielding closed-form solutions are used. It is possible to have several methods each of which generates only partial parameter vectors but which jointly solve the task. Finding SoR hypotheses reliably from point sets in real-time is challenging.

Many different methods exist for estimating initial cone or SoR parameters; the majority of them and in particular the fastest ones require normal vectors together with 3D points. Calculating a *dense* consistently oriented *normal vector image* therefore is a crucial first step. Sur-

face fitting and reconstruction from range data can be done without using normal vectors, which is not uncommon in the reverse engineering domain [XDQ04][HoK06]. Some other means of constructing a consistently oriented surface, such as a signed distance field, must then be chosen.

Hypothesis generation for an unknown number of objects profits much from a prior component labeling into *disjoint connected point sets* (regions, connected components) that are selected according to basic properties of curvature and shape. Some components serve as seed regions. Regions of interest created from previous images are also relevant, i.e. regions inside a (projected) bounding box that contains previously detected SoR and that is updated according to the motion estimate from the previous to the current image. Connected component labeling acts as an *attention* generator and effectively *factorizes* the point space for hypothesis generation. At the optimization stage, it limits the competition among candidate models for data and limits a point's uncertainty as to which primitive it belongs, if any.

Labeling coherent point regions is a form of range image segmentation on its own, but here it is a rather simple *precursory* step towards a different goal. The final point assignment follows the particular model fitting constraints and, in any case, differs from an initial partition created using general curvature properties. The expected size of the initial regions is also unclear: should rather small and homogeneous ones be strived for, with a tendency to over-segment the interest objects, or rather large ones, risking under-segmentation and inconsistency?

Hypothesis evaluation follows optimization and is necessary to check the validity of hypotheses modeling real objects with the stated properties. It has two closely related functions: rating and pruning the set of false-positive hypotheses, and estimating their geometric and topological relations. Thereby, some basic model knowledge is generated helping to decide if a hypothesis should be abandoned, processed again by a different method, or newly constructed by grouping components differently. This is another chicken-and-egg problem: effective hypothesis processing depends much on clear model expectations, which must themselves be generated bottom-up in a data-driven fashion - unless a model is directly available and suitable from an up-to-date building CAD description.

3.2.1 Parameter initialization: model or data redundant?

Hypothesis generation includes initializing the model parameters from selected data points. Two different strategies may be pursued. Hypotheses may be generated using the fewest data points that uniquely determine the model parameters, for example Marshall [MLM01], Beder and Förstner [BeF06] or Schnabel [SWK07]. Marshall et al. solve an equation system using only four non-collinear points with normal vectors to find the axis parameters of cone candidates. A single hypothesis is thereby efficiently generated, but a huge set of false model candidates may result (*model redundant* strategy). If seed points are selected randomly, an unavoidable dilemma results: the *closer* the points, the more will small errors, especially in the normal directions, boost model parameter errors, and the more *distant* the points the higher will be the likelihood of coming from different, unrelated primitives. Pruning the hypothesis set would be left to the optimization algorithm (EM) where hypotheses compete with each other. Because of the locality of optimization it is not at all clear that the better hypotheses will survive the earlier steps of this competition.

Model accumulation in a Hough space can be seen as an extreme kind of model redundancy: a *single* point generates a whole *subset* of primitives containing the point. For example, every point could lie on any cylinder whose axis is *orthogonal* to the point normal, therefore it votes for all axis orientations lying in a plane [Rab06]. A critical review of the Hough space approach is given in chapter 4. Hough space approaches are not primarily designed for real-time application, since their complexity is a polynomial in the number of points with a degree depending on the number of free model parameters.

The data redundant strategy, on the other hand, attempts to generate hypotheses from larger data sets than needed for model parameter estimation. Considerable initial effort is put into forming viable hypotheses, into consistency checking and robust fitting, but experiments have shown that this effort more than amortizes in the optimization phase. No great surplus of hypotheses will be generated and offered to the optimization compared to the number finally surviving. The likelihood of missed surface hypotheses may therefore be higher than for the model redundant techniques. In particular, when a model candidate fails in the optimization or verification stage and is discarded, relatively large portions of range data may be left unexplained if no other model is available to them.

3.2.2 Summary of alternatives

The following figure 3-4 refines the processing stages in figure 3-3 by providing different options which are investigated in detail in chapter 5. The little boxes inside the four main blocks are meant as basic building blocks that can and will be combined in different ways to form more complete algorithms. Choices at one stage also affect the amount of work spent at other stages. All choices were selected bearing in mind their potential to real-time operation.

Point feature mapping and connected components: Homogeneity criteria for point sets are investigated, including type classification by curvature sign, by magnitude (level of curvedness) or by using shape descriptors. To enhance the usefulness and stability of the partitions resulting from these homogeneity criteria, point operators may be performed at multiple scales. Another option is to provide direct manipulations on an initial component partition, using conditional morphology.

Hypothesis generation: Different methods are proposed to estimate direction(s) of rotational symmetry as a multi-linear axis structure, including new ones which, to our knowledge, have not been published before. Once the axis curve is known, the associated radius function can be estimated easily by a standard method. MIN/MAX chains, an extension to principal curvature directions, provide a simple way to estimate cylinder and cone axis directions. Operators to estimate points on the rotation axis from small neighborhoods are more generally applicable. The basic foot point transformation (FPT), the constrained (C-FPT), the directional (D-FPT), and the isotropic (I-FPT) FPT are proposed. Clouds of foot points seen as noisy curve (one-dimensional) representations are reduced to multi-lines by an adapted principal curve algorithm. A third group of algorithms estimate axis directions directly by casting the normal ray property (in section 3.1.1) into line geometric PCA minimization problems.

Hypothesis optimization: Designing the E-step probabilities or degrees of membership is one major task, in order to associate points with models. For the M-step (model fitting part), a

Levenberg-Marquardt optimization for cone parameters and an ICP-based optimization of pose parameters, applicable to general SoR, are investigated. Integrating the M-step and the E-step raises issues of converting different parameter representations. A special case of efficiently fitting a SoR as a 'left'/'right' extension to an existing SoR model is also handled.

Hypothesis evaluation and verification: It mainly contains various tests accumulating evidence of/against a hypothesis representing a real target object. Promising criteria that prevailed in the experiments include the *component explanation* (portion of initial point set explained by the final model), the *surface coverage* (portion of model surface area explained by assigned points), a *visibility* (viewing angle) criterion, the degree to which SoR objects are *tangential* to other, notably planar surfaces, and the degree to which *alternative hypotheses* explain the data, for example multiple intersecting planes or spheres. Further clues of validity are provided by the grouping relations, for example collisions (object penetration). As a result, decisions to accept or reject hypotheses or to retry hypothesis generation are available.

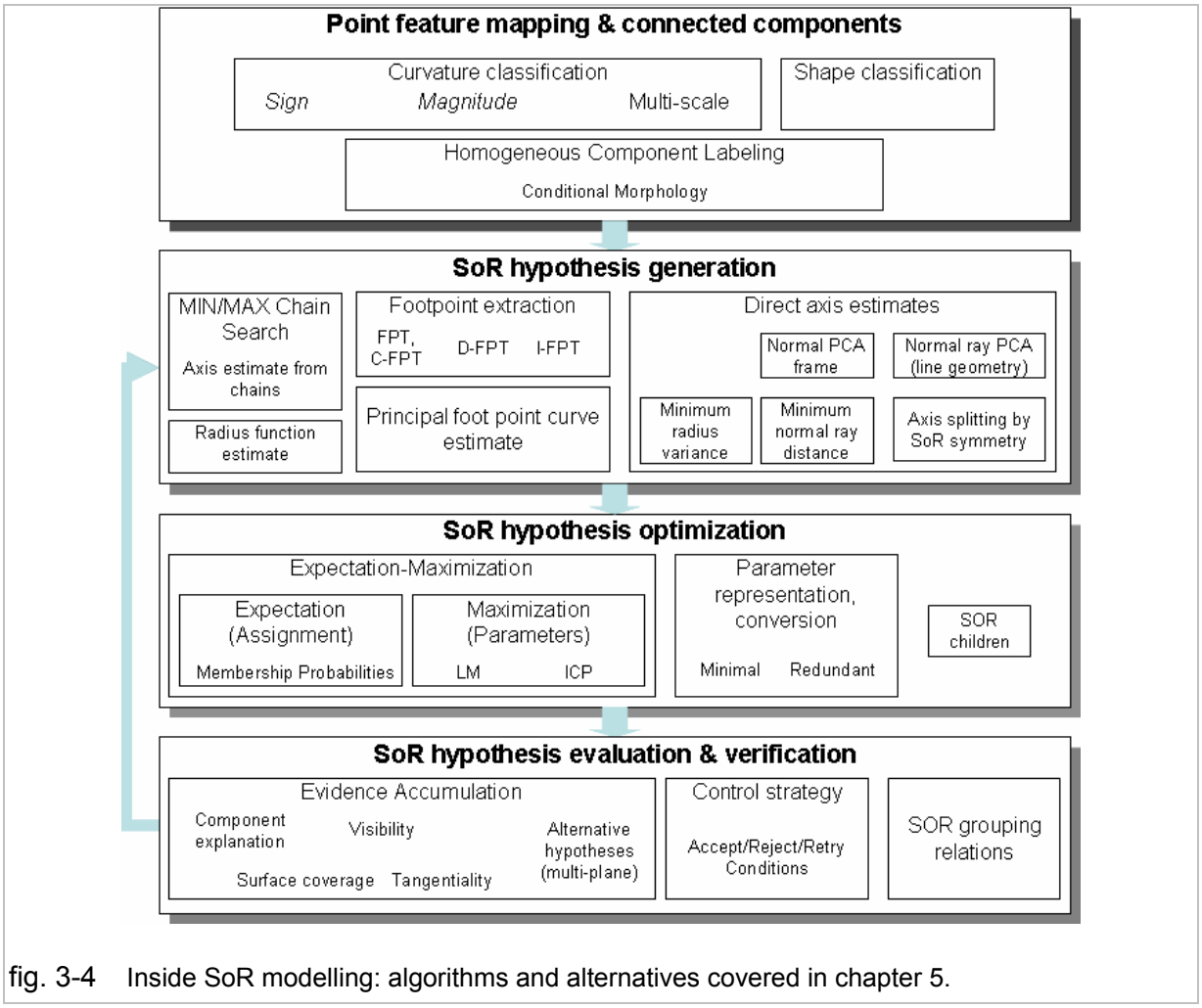

fig. 3-4 Inside SoR modelling: algorithms and alternatives covered in chapter 5.

4 Previous work

Generalized cylinders, surfaces of revolution and their subclasses have been widely used in solid modeling and in computer vision, e.g. in object recognition, for more than twenty years. Yet, their automatic and reliable reconstruction from point features remains challenging. Few algorithms exist for automatically synthesizing complex object structures in real time from occluded range views without a prior geometry model. Work from several problem areas is relevant: geometric representations and their properties, early vision and feature extraction from range images, segmentation and classification, statistical and algebraic methods for hypothesis generation, fitting methods, and nonlinear optimization techniques.

Pioneering work by Zerroug and Nevatia [ZeN96] addresses the recovery of 3-D shape descriptions from single *intensity* images and focuses on two subclasses of generalized cylinders with *planar* axes: their cross sections may either adopt any shape but be constant (PRCGC, 'planar right constant' GC), or be circular with a varying radius (called PRGC, 'circular planar right' GC). The theoretical focus is on imaging properties of such objects under orthographic projection that are invariant or quasi-invariant, i.e. change 'gently' under different viewing conditions. Parallel symmetric lines and 2D axis lines, especially the self-occluding contours of a GC image called *limbs* and the antagonist point pairs on them lying on the same cross section ('co-cross-sectional' segments) are considered. Topological properties, like junctions and cusps, and their possible combinations that may appear in an image of a single object due to self-occlusion are characterized as well. The practical part proposes a segmentation algorithm using complex heuristic correspondence search to detect and analyze those features, mainly parallel symmetries and ribbons. Recovery of a 3-D object model is possible under additional assumptions, such as cross sections at two ends with *different* orientations being visible (no *straight* axis!). Results on curved objects that seem to be well isolated suggest a remarkable accuracy; questions arise as to the robustness in an industrial setting, even with much simpler shapes, where occlusion and quantization noise are omnipresent. The running times reported seem to vary greatly.

A number of research papers focus on symmetry detection itself, as a tool for related problems like segmentation, registration, and matching [LoE06][PSG04][ZoL06][LZK08][DLP05]. The techniques may be distinguished by the class of symmetries considered where reflective or mirror symmetries have fewer parameters than rotational symmetries, the image or feature domain to detect symmetries in (2D perspective views versus 3D range images), the domain of the symmetry transform itself (2D image domain or 3D object domain), and, finally, the specific techniques and possible restrictions imposed on the scene views and the objects.

Loy and Eklundh [LoE06] study reflective and rotational symmetries by matching oriented and scale invariant feature points in intensity images. The symmetries themselves are mappings in the image plane, i.e. one region is similar to another one by mirroring at the 2D image axis or by rotating about an image point. The Planar Reflective Symmetry Transform by Podolak et al. [PSG04] capture the degree of symmetry of arbitrary shapes with respect to reflection through all planes in space This works for 3D images and 3D transformations, but mostly entire, isolated and non-occluded objects are shown. Zou and Lee [ZoL06] present an algorithm for detecting skewed rotational symmetry of 3D polyhedral objects from a 2D line

drawing. Li and Zhang [LZK08] detect reflective symmetries in man-made objects from perspective views in order to identify, segment, and track objects in video sequences in real time. They use dynamic programming and a Hough space approach for accumulating the symmetry parameters, and do not require prior object models. Deveau et al. [DLP05] extract revolution objects, mainly columns in architectural sites, from single laser scans. A Hough transform of the symmetry axis is computed from the range image contours. Candidate axes are refined by minimizing the root mean square error of the profile estimates, relying on clearly visible occluding contours (silhouettes) of the objects.

The following discussion will be devoted to range images. Characterizing local curvature and shape properties is often seen as an important precursor to segmentation and fitting, including revolution objects. Estimating curvature properties from range images reliably and efficiently has a long history; see Besl [BeJ88], Flynn [FlJ89], Koenderink [KVD92], Fisher [TrF95][CaF01], Jiang and Bunke's book [JiB97] and Petitjean's survey [Pet02] focusing on triangle meshes as summary resources. Tensor voting (Tang and Medioni [TaM99]) is a more recent development with similar goals and capabilities: extracting local surface and curve features such as junction points and point labeling using curvature properties. The tensor is equivalent to a 3x3 point covariance matrix together with its sorted eigenvalues whose ratio indicates the local shape (stick, plate, or ball). By tensor voting, tensors are aligned and accumulated in a neighborhood of a local feature, and a new 'saliency' tensor is produced. The method is best suited to assess feature saliency in *sparse* images, and to suggest smooth perceptual *continuations* between features. Despite linear complexity in the number of data points and in the neighborhood size, the running times are non-negligible. For mapping industrial facilities from dense images, extracting curves and junctions is of less importance, and would rather be performed after the fitting.

As to range image segmentation, from the discussion in section 3.2 it is obvious that only segmentation methods *combined* with model-(primitive-) based fitting are of interest. Lukács et al. [MLM01][LMM97][LMM98] present geometric fitting methods for quadratic surfaces such as spheres, cones, and tori, focusing on the high accuracy required for reverse engineering and quality inspection in manufacturing. They address degeneracy as smooth surface types become 'ambiguous', for example when their principal curvatures approach zero or become equal. One advantage of their fitting method is to handle various special cases of surfaces and automatically return the simplest type, without diverging as some parameter values - e.g. the curvature radius - become infinite. Another unique feature is the use of approximations to the Euclidian distance termed *faithful*. These are functions with the same zero set and the same derivatives at the zeroes as the original distance function but avoiding singularities when large but almost equal quantities are subtracted, for example. For nonlinear error minimization, the Levenberg-Marquardt procedure is used. Lukács [MLM01] estimates seed parameter values from a minimal number of data points. In the case of a cone, four points with normal vectors determine the rotation axis. A large number of segmentation experiments were performed with simulated and real range data, partly from public range image databases. Fitting errors are reported but no explicit figures on the accuracy of the *model parameters* (e.g. cone axis) with respect ground truth, or their sensitivity were found.

Faber and Fisher [FaF02] focus on least-squares fitting of general quadric surfaces, and compare the geometric distance to other distance measures (algebraic distance, Taubin's

distance, 3L-level sets). Methods are also discussed to mitigate the influence of outliers on the fitting results, applying an influence function of the distance instead of the distance itself. Accuracy and robustness, not predictable speed is the main concern.

Ahn and Effenberger [AhE03] address segmentation, outlier elimination, and model fitting in point clouds using a geometric error measure. A semi-automatic procedure 'Click-and-Clear' is proposed for the object recognition: the user selects a seed point in the point cloud and provides a low-level model type (e.g. a sphere). The algorithm initializes the parameters through model fitting around the seed point, and initializes the safety margin of a region of interest (ROI) around the current model feature, in order to exclude outliers and save computation time. The root mean square error (rms) σ of points inside the ROI is determined and points with distance below a fixed multiple $k\sigma$ denoted as inliers. The model is then fitted to the inliers; if the inliers' rms error becomes too large, the type of model is refined, i.e. some subclass is fitted and/or the safety margin is updated.

Sacchi in his PhD dissertation [Sac02] segments triangulated point models into parts corresponding to geometric primitives, in particular planes, spheres, cylinders, cones, and tori. He uses region growing with a homogeneous shape criterion, and proposes directional curvature estimates for triangulated data, using nine pairs of points with attached normal vectors. The different adjacency relations in triangulated data (vertex-vertex, vertex-triangle, and triangle-triangle) are all exploited to interpolate normal vectors and to construct the value pairs. Initial estimates of primitive parameters, e.g. cones, are generated bottom-up from one or two triangles. The segmentation goal function is optimized using an Evolutionary Algorithm which takes unpredictable running times. Some over-segmentation of cylinders and cones remains because of the sensitive seed parameter generation.

A somehow related problem is the *broken-pottery puzzle* [OrW00][WiC04][Wil04]: assembling *shards* from a broken object, so as to reconstruct its smooth shape as a single *surface of revolution*. The axis and profile curve of the vase or pot needs to be estimated from each piece, existing as a 3D point data set, and also from configurations of several pieces. Of course, the broken-pottery problem exhibits specifics that do not apply to vessel reconstruction in industrial sites: the entire object has a straight axis, its profile curves are smooth but of complex (artistic) design, and the shards are *disjoint* patches with matching boundary curves where they broke apart. Orriols et al [OrW00] focus on the single shard case; the unknown axis and profile curve parameters span the so-called latent space. Conical surfaces form the primitives from which general surfaces of revolution are built; the object's shape model is a Bayesian combination. Complex profile curves are decomposed top-down into linear segments using mixtures of probabilistic principal component analyzers (a Gaussian mixture density distribution). The number of line segments must be known, and it has an impact on the axis and profile estimation results. An EM algorithm is then used to optimize the parameters. The approach requires no curvature information to characterize different surface types.

Willis [WiC04][Wil04] tackles the complex multi-view matching and registration problem with multiple shards, using a Bayesian maximum likelihood formulation to estimate the global SoR parameters (axis, profile curve), the individual pose transformations for each shard, and the matching break curves for all pairs of adjoining shards. (In our case, SoR are reconstructed from single views; multi-view registration and merging is handled at the higher level of SLAM

mapping where SoR form just one example class of features to be matched.) To master the search complexity, shard configurations are incrementally extended by appending the most probable pair-wise configuration containing an unused shard to a set of promising configurations which are kept at any time. Profile curves are estimated using implicit polynomials up to degree six. Normal vectors of shard data points are only used on the boundaries.

Estimating SoR parameters from shards requires an initialization of the axis direction which is performed by an algorithm due to Pottmann and Randrup [PoR98]. They use points with normal vectors to calculate a velocity field - the velocity vector is orthogonal to the normal vectors - and obtain the velocity parameters as a 6D generalized eigenvector of a quadratic form. The method asks for prior knowledge which points belong to which class of motion because the general algorithm may face numerical instability if applied to points that fit well to a more specialized case, i.e. cone or cylinder. Estimating a rotation axis from a piece (shard) is generally - i.e. independent of the method used - difficult and potentially unstable if

- the visible patch only subtends a small angle about the axis, or
- the principal curvatures have similar magnitude (spherical, ellipsoidical, or saddle-type).

Taylor's dissertation [Tay04] develops fundamental visuomotor skills for a humanoid robot to perform manipulation tasks in an unstructured home (household-type) environment. A broad scope of problems is tackled, including hand-eye coordination and calibration, visual servoing, object tracking using multi-sensory cues, high level path planning and task planning. Relevant contributions are also made to the perception aspect, tangibly classification, and fitting of *geometric primitives* to *range images*. The skills are integrated to perform domestic tasks like locating and grasping a previously unknown object, or pouring the contents of an interactively selected cup into a bowl.

The robot acquires a dense 3D map of its workspace using a stereoscopic light stripe range scanner. I.e., a laser diode casts a (rotatable) vertical light plane onto the scene and the resulting line stripe is observed by a stereo camera system to recover a dense range map. To identify objects of interest, a geometric world model is constructed. After estimating point normals and curvatures, the range image is segmented into patches using a split-and-merge algorithm, and their shapes are characterized by analyzing the normal vector distributions arranged in a *Gaussian image* [Hor84]. Sufficiently large patches of homogeneous surface type can be fitted to geometric primitives (planes, cylinders, cones and spheres). Boxes, balls, bowls, cups and cans are identified as generic arrangements of the extracted primitives. An attributed scene graph is constructed, and convexly connected sub-graphs are matched to corresponding convex model objects. A relationship to our work lies in

- Using a *minimal* parameterization of primitives, and using Levenberg-Marquardt (LM) for parameter optimization, and
- Using data redundancy (homogeneous patches) to generate initial parameter estimates. The method of parameter estimation is similar to the basic foot point transformation in section 5.3.

No model-driven region growing (expectation step, EM) is provided. Also, the entire scene analysis is performed only once; for manipulation the robot switches to a more efficient tracking mode. The objective [Tay04] is not to generate a 3D environment map in real time.

Rabbani's work [Rab06][Rab07] presents methods for the 3D modeling of existing *industrial installations* using *point clouds* and intensity *images*. To our knowledge, it marks the highest level of automation achieved to date in this application domain. The techniques were applied at a large scale in petrochemical plants. In the dissertation [Rab06] a multi-stage algorithm for *unorganized* 3D point clouds is described. The input data are first segmented using nearest-neighbor search into mutually disjoint and smoothly connected regions, the smoothness criterion assessing *local* normal similarity and *local* spatial connectivity. Surface curvature is inferred indirectly from the *residual error* of plane fitting, which depends on noise and scale (sampling density). The grouping into regions is guided so as to *avoid over-segmentation at the cost of under-segmentation* which, however, may fall short of shape *homogeneity*.

Planes and *cylinders* are automatically detected using a *Hough transform*, which is applied to the point set of each segmented surface patch. A two-step, cascaded procedure is applied avoiding a 5D Hough space for cylinder representation. The first step builds a 2D Hough space for the axis orientations, and the second step determines the according radii and positions for each best candidate selected in step 1, searching in a 3D Hough space. During the subsequent hypothesis verification different types of objects, for example planes and cylinders are simultaneously detected on overlapping data domains, as no prior curvature or shape classification is available. For ambiguity resolution, planarity and cylindricality tests are proposed, analyzing the histograms of residual distances, angle variations, and aspect ratios. Planar patches are expected to have peaked distance and angle histograms. Cylinders cover a broad range of normal angles (visibility sector > 90° or at least 25%); they must exceed a length threshold and their radii lie within given bounds [0.05m, 1m].

To construct a Hough space of cylinder orientations, each data point votes for *any* cylinder with axis *orthogonal* to the point *normal*. These orientations span a plane and, being unit vectors, are located on a Gaussian sphere. The locus of cylinder orientations forms a *great circle* on this sphere. A point with different normal vector votes for a different great circle of candidate cylinder axes. If both points lie on the same cylinder, its axis direction is therefore found in the *intersection* of two great circles. By tesselating the Hough sphere into polar coordinate cells of equal area (!) and accumulating the votes for each cell, cylinder orientations with maximum support are thus found. This step does not need any grouping or segmentation of points.

In practice, despite its elegance and simplicity, the Hough approach has its difficulties. First is the computational cost: a primitive with p parameters (a cylinder with $p=5$ is a simple case) leaves $p-1$ *degrees of freedom* for each point. By dividing all value ranges into B bins, $N \cdot B^{p-1}$ votes will be generated for N points, not to mention the overhead of selecting, testing and validating the best hypotheses. This is not feasible for real-time mapping. Secondly, finding maximal votes is susceptible to *local (false) maxima* because the orientation distribution is smoothed (blurred) when scanning at low resolution, say, on a complex pipe with flanges or a facetted (prismatic) cylinder. The problem may be exacerbated when, in the cascaded approach, each stage relies on the hypotheses selected at earlier stages.

Since the model parameters are *quantized* in the Hough space, a more accurate model fitting step follows. For fitting models to point clouds like planes, cylinders, spheres, cones and tori, a nonlinear optimizer (LM) minimizing the orthogonal distance is chosen. Quadric fitting based on the algebraic distance yields the starting values. [Rab06] also proposes a procedure for fitting *composite* (CSG) models to data. This, in general, admits only approximate measures of shortest distance three of which are compared regarding speed and accuracy. Derivatives of the error function must be approximated by finite differences. Constraints from the CSG construction are specified as equalities or inequalities of directional angles, translation magnitudes or sizes, and can be met in two ways. Either an unconstrained minimization is performed and the constraints are enforced after each loop of iteration, or a constrained nonlinear optimizer with Lagrange multipliers is applied. Prior knowledge from the CAD plant design is required as to which primitives and constraints define an object, e.g. a flanged T-junction, and how to partition and allocate the sample data.

As range data from several viewpoints must be captured, an automatic scan registration is also needed [Rab06]. Two different methods are presented for this purpose, an 'indirect' method for getting approximate transformation parameters from matching model parameters, and a 'direct' method refining all transformation and model parameters from a number of range images simultaneously (integrated bundle adjustment). Corresponding objects for scan registration are found by constraint propagation. Unlike our incremental SLAM scenario, all range views are available at once.

5 Technical developments

Chapter 5 is organized as follows; the reader may refer back to the summary section 3.2.2 with the block diagram in figure 3-4. Section 5.1 covers point feature mapping and connected component labeling. Sections 5.2-5.4 present alternative methods for hypothesis generation, starting with the heuristic and historical curvature chain search (section 5.2). Approximations of the unknown axis by point clouds (foot point transformations) follow in section 5.3. Fast algorithms to estimate the axis in closed are covered in section 5.4. Section 5.5 deals with the parameter optimization by EM. Hypothesis evaluation and verification are covered in sections 5.6 focusing on individual SoR hypotheses and in 5.7 on relations that generate new information for hypothesis evaluation. Section 5.8 puts the pieces together and proposes control strategies to combine the basic methods to complete SoR algorithms.

5.1 Dense point feature maps

The first sub-goal is to decompose range images into topologically connected regions some of which form seed regions for SoR hypotheses. These regions are point (address) sets with scan order information; they provide direct access to the hypothesis generation and are themselves computed with constant overhead per point. Elementary point features, i.e. normal vectors, edge, curvature or shape type are used to achieve this goal.

5.1.1 Fast normal estimation

Each range view (≈ 60-80000 3D points) should be densely covered by normal vector estimate. These time consuming calculations are indispensable because normal vectors form the basis for many steps to follow, such as estimating curvature and shape or generating SoR axis hypotheses.

The simplest and fastest method is to calculate a normal as the cross product of two difference vectors to the row and column neighbors, but it is too sensitive to estimate curvatures. Instead, plane fitting is applied to the points in a mask M of size $n \times n$ around every point [FIJ89]. The normal is - up to its sign, which is chosen so as to point towards the sensor - the eigenvector to the smallest eigenvalue of the 3x3 covariance (scatter) matrix

$$\mathbf{S} := \sum_{\mathbf{p} \in M} (\mathbf{p} - \mathbf{c})^T (\mathbf{p} - \mathbf{c})$$

where $\mathbf{c}$ denotes the mean (center) of the mask. Our implementation keeps the overhead per point constant, independent of the mask size n, by updating and down-dating the matrix as a sliding window:

- M_{ij} denotes an $n \times n$ mask (window) of points centered at point address (i - $n/2$, j - $n/2$)
- C_{ij} denotes an $n \times 1$ column mask (window) centered at (i - $n/2$, j).

Moving to the next point $\mathbf{p}_{ij}$ in scan row i and column j requires two sliding window operations on the current mask as illustrated in figure 5-1:

$$C_{ij} = C_{i-1,j} \,\Theta\, \mathbf{p}_{i-n,j} \oplus \mathbf{p}_{i,j}$$
$$M_{ij} = M_{i,j-1} \,\Theta\, C_{i,j-n} \oplus C_{i,j} \qquad (5.1)$$

where the update ('⊕') respectively down-date operators ('Θ') for a point $\mathbf{p}$ change the mask center $\mathbf{c}$ and covariance matrix $\mathbf{S}$ as follows; the sum '+' being taken for updating '⊕', and the difference '-' for down-dating 'Θ':

$$\mathbf{c}_{M\{\oplus|\Theta\}\mathbf{p}} := \mathbf{c}_M \pm \frac{1}{n \pm 1}(\mathbf{p} - \mathbf{c}_M) \qquad \mathbf{S}_{M\{\oplus|\Theta\}\mathbf{p}} := \mathbf{S}_M \pm \frac{n}{n \pm 1}(\mathbf{p} - \mathbf{c}_M)\cdot(\mathbf{p} - \mathbf{c}_M)^T$$

Updating '⊕' respectively down-dating 'Θ' a mask M_1 by another mask M_2 or a column vector as in the second line of (5.1) act on the center $\mathbf{c}$ and the covariance matrix $\mathbf{S}$ as follows; again, the sum applies to updating and the difference to down-dating:

$$\mathbf{c}_{M_1\{\oplus|\Theta\}M_2} := \frac{n_1}{n_1 \pm n_2}\mathbf{c}_1 \pm \frac{n_2}{n_1 \pm n_2}\mathbf{c}_2 \qquad \mathbf{S}_{M_1\{\oplus|\Theta\}M} := \mathbf{S}_1 \pm \mathbf{S}_2 \pm \frac{n_1 n_2}{n_1 \pm n_2}(\mathbf{c}_1 - \mathbf{c}_2)\cdot(\mathbf{c}_1 - \mathbf{c}_2)^T \qquad (5.2)$$

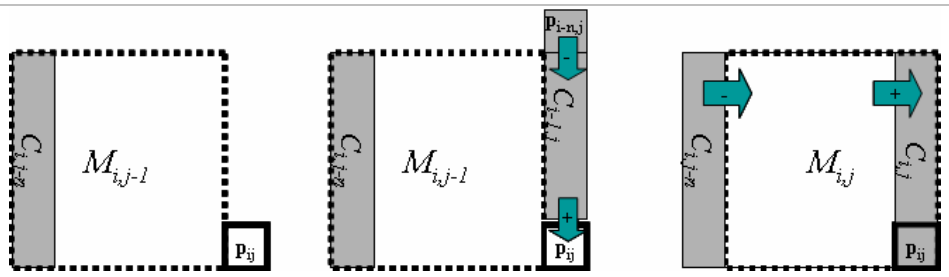

fig. 5-1 Sliding window operation for normal estimation

For calculating the smallest eigenvalue and eigenvector of the symmetric positive definite matrix $\mathbf{S}$, it turns out fastest to solve the 3rd degree characteristic polynomial for its smallest root λ_1 - the solution equation directly reveals which one is smallest - and form the corresponding eigenvector as being orthogonal to two linearly independent columns of $\mathbf{S} - \lambda_1 \cdot \mathbf{I}_3$; see Kopp's algorithm [Kop06] for details. Using here the vector product works only in $\Re^3$ but is faster than even an incrementally updated Cholesky decomposition that might also be applied to solve for the eigenvectors. The rationale behind this optimization is numerical library routines such as MATLAB (*eig*, *eigh*, *eigv*) being optimized for high-dimensional matrices and applying them to many tiny (3x3) problems incurs some overhead.

Our sliding window technique with the MATLAB routines substituted by Kopp's algorithm gave a speed-up factor of 10 to 20 for average normal mask sizes of 7 to 9. Slight losses of numerical accuracy were observed only where two nearly identical smallest eigenvalues exist, i.e. where the point cluster is locally more 'stick-like' than 'plate-like', mostly near prominent crease or jump edges and near the image borders.

5.1.2 Edge point detection

Surfaces of revolution are supported by smooth seed regions, whereas *edge points* violate the smoothness and should be excluded beforehand. Jump edge points in a range image over a real-valued parameter space $\Re^2$ are defined as discontinuities of distance, and crease

edges as discontinuities of normal vectors, i.e. large changes of normal directions occurring within arbitrarily close point distances [JiB97]. For sampled range data, the definitions are in practice replaced by difference values and thresholds, i.e. percentiles in distance histograms.

As to jump edges, it proved useful generalizing the notion to include *strong local variations* of the *sampling density*, which counts the number of points per unit surface area. Regions without any range measurements ('holes') have zero sampling density, and their neighbor points are considered as edge points. True jump edge points with large distance variations in a small neighborhood cover a large area as well and therefore have a low sampling density. The local surface area covered by a point $\mathbf{p}_{ij}$ is coarsely estimated from the 4-neighborhood, and the sampling density D_{ij} is the reciprocal:

$$D_{ij} = 4/\left(A_{i,j}+A_{i-1,j}+A_{i,j-1}+A_{i-1,j-1}\right) \quad where \quad A_{i,j} := \left\|\left(\mathbf{p}_{i+1,j}-\mathbf{p}_{i,j}\right)\times\left(\mathbf{p}_{i,j+1}-\mathbf{p}_{i,j}\right)\right\|.$$

If one of the points referred to is undefined, D_{ij} is set to zero. The *actual* sampling density D_{ij} related to the *expected* density at the distance of point $\mathbf{p}_{ij}$ gives the normalized density, assuming the sensor origin at $\mathbf{0}$, and a uniform angle grid with resolutions $\Delta\rho$, $\Delta\alpha$ for simplicity

$$\overline{D}\left(\mathbf{p}_{ij}\right) = D_{ij}\Big/\left(\left\|\mathbf{p}_{ij}\right\|^2 \cdot \sin\Delta\rho\cdot\sin\Delta\alpha\right)$$

In the histogram of normalized density values a low threshold value corresponding to a 5-10% percentile is selected to mark jump edge, respectively, low density edge points.

Similarly, points with high normal variation called crease or roof *edge points* are detected. Some percentile *p* in the histogram of crease edge strength values is chosen as a threshold. A simple and normalized measure of crease strength was devised that is independent of curvature calculations. At a point $\mathbf{p}$, the maximum *directional* crease strength is taken along four lines passing through $\mathbf{p}$ in four directions $d\in\{0, 1, 2, 3\}$ of the 8-neighborhood. Essentially, the crease strength $crs(\mathbf{p}, d, \mathrm{n})$ at point $\mathbf{p}$ is related to ($\sin^2()$ of) the *angle* between the *average normals before* and *behind* $\mathbf{p}$, taking at most n steps forward in direction d, and backward in direction $(d+4) \bmod 8$, where crease angles >90° have a *csr*-value of 1.

$$crs\left(\mathbf{p},d,n\right) := \begin{cases} 1-\dfrac{\left(\mathbf{n}_{d\rightarrow}^T\cdot\mathbf{n}_{\rightarrow d}\right)^2}{\left(\mathbf{n}_{d\rightarrow}^T\cdot\mathbf{n}_{d\rightarrow}\right)\left(\mathbf{n}_{\rightarrow d}^T\cdot\mathbf{n}_{\rightarrow d}\right)} & if\ \mathbf{n}_{d\rightarrow}^T\cdot\mathbf{n}_{\rightarrow d}\geq 0 \ \in[0,1] \\ 1 & else \end{cases} \qquad (5.3)$$

$$where \quad \mathbf{n}_{d\rightarrow} := \frac{1}{n_d}\sum\nolimits_{j<n}\mathbf{n}_{d,j}, \quad \mathbf{n}_{\rightarrow d} := \frac{1}{n_{d\oplus 4}}\sum\nolimits_{j<n}\mathbf{n}_{d\oplus 4,j}$$

Crease strength at a point $\mathbf{p}$ using n neighbor points is the maximum value $crs(\mathbf{p}, d, n)$ for $d = 0, 1, 2, 3$, and the crease direction is the direction maximizing (5.3), orthogonal to the running direction of the edge contour itself. The crease strength image is used in two ways. Firstly, a crease strength histogram is formed and an upper percentile (90-95%) value is chosen as a threshold to mark non-smooth point areas. Secondly, when actually *edge points* are needed, only the local maxima in the crease strength map are retained (non-maxima suppression).

5.1.3 Directional curvatures

Some algorithms for detecting rotationally symmetric primitives require knowledge of the minimal and maximal curvature at each point, i.e. sign and magnitude of the directional curvatures. A simple and widely used approximate formula due to Jain [JiB97] estimates the directional curvature between two points $\mathbf{p}$, $\mathbf{p}'$ with the respective unit normal vectors $\mathbf{n}$, $\mathbf{n}'$ by expanding the surface in the normal direction by unit length and relating the distance between the 'expanded' points $\mathbf{p}+\mathbf{n}$, $\mathbf{p}'+\mathbf{n}'$ to the original point distance. With $\Delta\mathbf{p}$:=$\mathbf{p}$-$\mathbf{p}$', $\Delta\mathbf{n}$:=$\mathbf{n}$-$\mathbf{n}$':

$$k(\mathbf{p},\mathbf{p}') \approx \operatorname{sgn}\left(\|\Delta\mathbf{p}+\Delta\mathbf{n}\|-\|\Delta\mathbf{p}\|\right)\cdot\frac{\|\Delta\mathbf{n}\|}{\|\Delta\mathbf{p}\|} \qquad \text{(Jain's formula, adapted from Bunke [JiB97])} \qquad (5.4)$$

This formula is reasonably fast to compute and often yields acceptable results. However, between *close* points the normals need to be scaled to a length $l<\|\Delta\mathbf{p}\|/2$ to discern a concave crease from a convex one, see figure 5-2a for illustration. Scaling gives biased results unless performed with a uniform factor on the entire image. The bias can be avoided by taking the *limit* as the length $l\to 0$ of a scaled curvature function $k(\mathbf{p}, \mathbf{p}', l)$ defined as follows:

$$k(\mathbf{p},\mathbf{p}',l)=\frac{\|\Delta\mathbf{p}+l\cdot\Delta\mathbf{n}\|-\|\Delta\mathbf{p}\|}{l\cdot\|\Delta\mathbf{p}\|}=\frac{\sqrt{\Delta\mathbf{p}^T\Delta\mathbf{p}+2l\cdot\Delta\mathbf{p}^T\Delta\mathbf{n}+l^2\Delta\mathbf{n}^T\Delta\mathbf{n}}-\|\Delta\mathbf{p}\|}{l\cdot\sqrt{\Delta\mathbf{p}^T\Delta\mathbf{p}}}=:\frac{f(l)}{g(l)} \qquad (5.5)$$

By l'Hospital's rule, since $f(0) = g(0) = 0$ and f and g are differentiable around 0:

$$\lim_{l\to 0}k(\mathbf{p},\mathbf{p}',l) = \lim_{l\to 0}\frac{\dot f(l)}{\dot g(l)}=\lim_{l\to 0}\frac{\Delta\mathbf{p}^T\Delta\mathbf{n}+l\cdot\Delta\mathbf{n}^T\Delta\mathbf{n}}{\sqrt{\Delta\mathbf{p}^T\Delta\mathbf{p}+2l\cdot\Delta\mathbf{p}^T\Delta\mathbf{n}+l^2\Delta\mathbf{n}^T\Delta\mathbf{n}}\cdot\sqrt{\Delta\mathbf{p}^T\Delta\mathbf{p}}}= \frac{\Delta\mathbf{p}^T\Delta\mathbf{n}}{\Delta\mathbf{p}^T\Delta\mathbf{p}}\in\left[-\frac{1}{\|\Delta\mathbf{p}\|},\frac{1}{\|\Delta\mathbf{p}\|}\right] \qquad (5.6)$$

The dot product formula (5.6) avoids the bias due to normal scaling, and is more efficiently computable than the original one (5.4). On a simple geometry like a *circular* cross section of a convex or concave cylinder (figure 5-2b), reasonable values are obtained: since the vectors $\Delta\mathbf{p}$ and $\Delta\mathbf{n}$ are linearly dependent (parallel) with a proportionality factor $r>0$: $\Delta\mathbf{p}=\pm r\cdot\Delta\mathbf{n}$ where '+' applies to a convex and '-' to a concave cylinder surface, we get, as expected:

$$k(\mathbf{p},\mathbf{p}')\approx\frac{\Delta\mathbf{p}^T\Delta\mathbf{n}}{\Delta\mathbf{p}^T\Delta\mathbf{p}}=\pm\frac{1}{r}$$

Formula (5.6) is symmetric in both points between which a directional curvature is desired, using the *difference* of their normal vectors. Another alternative could have been the slightly different formula for directional curvature by Taubin [Tau95] which considers only the one normal $\mathbf{n}$ at the reference point $\mathbf{p}$ (to note that the '-' sign is due to our definition $\Delta\mathbf{p}$:=$\mathbf{p}$-$\mathbf{p}$'!):

$$k^{Taubin}(\mathbf{p},\mathbf{p}')\approx -2\frac{\Delta\mathbf{p}^T\mathbf{n}}{\Delta\mathbf{p}^T\Delta\mathbf{p}}$$

The comparative accuracy of both approximations has not yet been investigated.

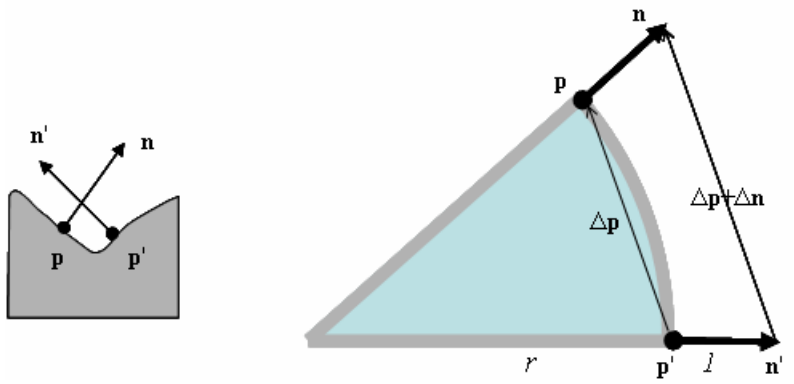

fig. 5-2 a) Normal scaling needed in the directional curvature formula (5.4) (left), b) Curvature approximation (5.6) illustrated at an example of a circular cross section (right).

5.1.4 Multi-scale curvature classification

For each point, directional curvatures (5.6) to all neighbor points with chessboard distance m_{dc}, lying on a square border, are calculated. Curvatures to the neighbors in opposite directions d, $(d+4) \bmod 8$ are averaged to get the curvature for one direction $d = 0\ldots3$. Coarsely quantized principal curvatures κ_{min}, κ_{max} are estimated as the minima and maxima of the $4m_{dc}$ directions for each point. Calculating directional and principal curvatures in this way is the only pre-processing step incurring linear and not constant overhead in the mask size. To limit the quantization errors, m_{dc} should be chosen ≥ 2, i.e. the mask size $n = 2m_{dc}+1 \geq 5$.[4]

5.1.4.1 Curvature type

Mean curvature H and Gaussian curvature K are obtained from the standard formulae [JiB97]

$$H = (\kappa_{min} + \kappa_{max})/2, \quad K = \kappa_{min} \cdot \kappa_{max}$$

enabling surface curvature classification, for example using

- Curvature *type* - the 8 possible sign combinations of Mean and Gaussian curvature [FIJ89] henceforth referred to as HK classification
- Local *shape* descriptors such as the one introduced by Koenderink [KVD92]
- Curvature *magnitudes* (levels) as suggested by Jagannathan [Jag05].

In any case, we wish to discern candidate regions from planar or low-curvature regions and from high curvature regions representing chamfers between planar patches, thin objects like poles, rods, or cables, and highly cluttered areas. Seeking connected components with low

[4] On range images with a high spatial resolution, larger mask sizes may be needed. We considered forming an image pyramid of directional curvatures, repeatedly doubling the distance and approximating directional curvatures using intermediate points at half the distance. Thus, the bulk of work would become $O(\log m_{dc})$. This scheme has not been implemented; a mask size of 9 proved sufficient in all cases at tolerable running times.

variations of shape within and high variation between the clusters is a classification problem where the number of classes and even the precise criteria are unknown, as the object types of interest (having some rotational symmetry) do not correspond one-to-one to any known curvature type or shape descriptor value. The problem may also be seen as a consistent labeling problem. We looked at stochastic relaxation and Associative Markov networks [Ang05] offering expensive integer programming solutions. The labeling problem is similar and an important precursor to range image segmentation. In our case, we seek a seed region finder towards a final goal which is SoR modeling and *not* curvature classification. No 'high-end', oversized effort towards a sub-goal should distract from the final goal.

The naive and fastest way of HK classification, using fixed thresholds $\pm\varepsilon_H$, $\pm\varepsilon_K$ of Mean resp. Gaussian curvature, fails because the thresholds need to be adjusted to different objects, varying from one range view to the next during exploration. Moreover no threshold for single-point curvature will work. Due to varying sampling density and quantization errors, different types of curvature may be interfused even on a perfectly smooth looking object. E.g. on a convex cylinder not only RIDGE, but many FLAT and even VALLEY points occur. The problem has been recognized and addressed a long time ago, for example by Fitzgibbon et al. [FEF97] proposing hysteresis thresholding, but has not been completely solved.

For SoR mapping, curvature histogram analysis at multiple scales was performed which is fast enough to be carried out in real time as the effort to build and analyze histograms depends mainly on the bin subdivision (200-400 bins in our experiments). Each curvature histogram is partitioned into local frequency maxima, called modes. A mode is characterized by its *dominance*, calculated from its total normalized *frequency* and its *span*, the minimum relative height difference to neighboring minima. As many spurious peaks will generally appear due to noise, the histograms are smoothed by a one-dimensional Gaussian kernel of size 3-5. Kernel sizes ≥5 however spread out and blur the modes too much.

The modes represent different curvature levels, and the minima between them serve as thresholds to separate the value domain into intervals, maximizing the expected curvature distances between and minimizing the distances within classes, i.e. minimizing the chance that a random sample is misclassified. The minima that separate the mode containing the value 0 from its right and left neighbor modes are used as thresholds ε_+, ε_-, respectively, to delineate low-curvature regions. If the bin containing 0 does not belong to any mode, the adjacent left or right mode is taken, whichever is more dominant. To find a border in case that a mode has no right or left neighbor, its peak value is connected to the rightmost resp. leftmost value, and the value of minimum (negative) distance from this line is taken as the interval bound.

Mode partitioning of histograms representing *single point* curvatures (at scale k=0) is not sufficient, since it pays no regard to *spatial connectivity*: the points captured in one histogram mode may not form connected regions. Figure 5-3 shows a curvature histogram with three modes centered about values a_1, a_2, a_3, respectively. The corresponding image has two connected regions R_1 and R_2 where R_1 has points with two curvature values $\approx a_1$ and $\approx a_2$ interfused, and R_2 contains values a_2 mixed with a_3. No mode-separating thresholds ε_1 and ε_2 therefore partition the points into homogeneous regions. At scale $k>0$, by averaging curvature within masks of size 2^k, the histogram will contain two modes with peak values between a_1

and a_2 for M_1 and between a_2 and a_3 for M_2. The two modes now represent disjoint regions, and are separable by a minimum since a mixture of values a_1 and a_3 is less likely.

Mode partitioning is therefore repeated at several scales each of which yields scale-specific thresholds. A point is classified as a low curvature candidate, for example, only if its curvature lies inside the threshold interval at each scale.

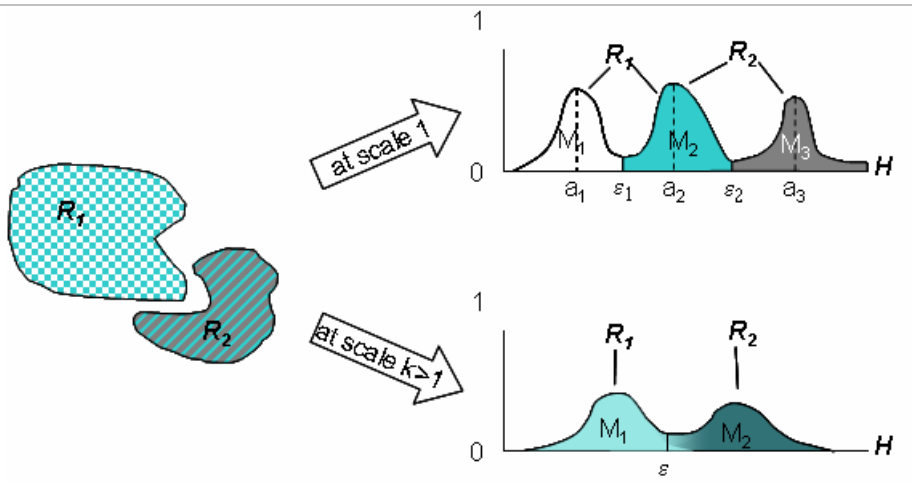

fig. 5-3 Curvature histogram produced at different image scales

5.1.4.2 Levels of Curvedness

Besides characterizing surface shapes, a fast and coarse classification of objects in a scene view is possible by curvature *magnitude*. The goal is to distinguish smooth vessels or pipes of various kinds (medium curvature magnitudes) from pillars, beams or clutter (high curvature) and from almost flat areas (low curvature). Measures of *curvedness* or *bending energy* are abundant in the literature; Jagannathan e.g. [Jag05] uses the following measure on surfaces represented by triangle meshes. All curvature and shape measures in this section apply to 3D points but the point argument is omitted.

$$CM = \sqrt{\kappa_{min}^2 + \kappa_{max}^2} \tag{5.7}$$

Evaluation of the measure 5.7 on scene views from the THERESA plant typically yielded an overwhelming majority of low-curvature, and only a tiny fraction 0.2-0.3% of high curvature points, i.e. an inconvenient curvedness distribution to analyze. The crease edge strength $crs()$ (5.4) assessing normal variation, a form of curvedness as well, produced more informative histograms. (This problem with CM might as well be relieved by taking the logarithms, as originally proposed by Koenderink [KVD92]).

The interval of crease strength values $[0,\ CRS_{max}]$ can be partitioned into a user-specified maximum of $k+1$ *levels of curvedness* by choosing k separator values $CRS_1 < \ldots < CRS_k$ as described in the previous subsection. After mode partitioning, the k most dominant modes are extracted. Two adjacent dominant frequency maxima are separated at the one of several minimum frequency bins where the resulting sample variance would become minimal.

After assigning curvedness levels to all points, a local homogeneity criterion is implemented, using the absolute differences of curvedness levels in its 8-neighborhood. A point lies in a

homogeneous region if both the sum and the maximum of the level (integer) differences remain below some limit (default values: 9 for the sum of differences, 3 for the maximum). In this way, the homogeneity criterion tolerates gradual changes in the curvedness level inside each region. Next, a component labeling algorithm (see section 5.1.5) extracts the connected regions using this homogeneity criterion, and calculates a *mean curvedness level* for each *region*. Regions with levels of curvedness excluding the lowest and the highest ones, and having a low Gaussian Curvature $|K|\approx 0$ are most eligible for SoR hypothesis generation.

5.1.4.3 Spatially adaptive shape classification

Regarding shape, the HK classification has one significant shortcoming: the thresholds, although calculated automatically for each scene view, still need to capture all objects within one view. A mechanism other than a threshold of Gaussian curvature is needed to better distinguish truly conical shapes (ridge, valley) from adjacent elliptical (peak, pit) or hyperbolic shapes (saddle ridge, saddle valley), where 'adjacent' could mean either *spatially* adjacent structures in Euclidean space, or *similarity* of *shapes*. In particular, a measure was looked for taking the *ratio* between minimum and maximum curvature magnitude into account. Koenderink's shape index [KVD92] does precisely this: it characterizes shape types by one *real value* SC:

$$SC = -\frac{2}{\pi}\tan^{-1}\left(\frac{\kappa_{max}+\kappa_{min}}{\kappa_{max}-\kappa_{min}}\right)\in[-1,1] \tag{5.8}$$

ranging from -1 ($\kappa_{min}\approx\kappa_{max}<0$ - pit, 'spherical cup') to 1 ($\kappa_{min}\approx\kappa_{max}>0$ - peak, 'spherical cap') via mirror-symmetric intervals of $[-5/8,-3/8]$ ($\kappa_{min}<0$, $\kappa_{max}\approx 0$ - valley, called 'rut') and $[3/8, 5/8]$ ($\kappa_{min}\approx 0$, $\kappa_{max}>0$ - ridge), joined by a center interval $[-3/8, 3/8]$ ($\kappa_{min}\approx -\kappa_{max}$ - hyperboloids).

Cantzler and Fisher [CaF01] compared the shape classification by SC with the HK classification and showed that the former is more stable at low thresholds on scenes containing cylinders, and can deal better with noise in images which contain different types of surfaces. One drawback of the SC method, namely eq. 5.8 being undefined on flat surfaces and giving unstable results on noisy flat surfaces is avoided by using the curvedness levels (5.1.4.2) to classify flat regions beforehand and independently.

In this case study, we need to go one step further: providing a *regionalized* shape classification mainly for conical shapes. To find region-specific automatic thresholds $\varepsilon_{Cone\leftrightarrow Ell}$, $\varepsilon_{Cone\leftrightarrow Hyp}$ discerning genuinely conical parts from more elliptical and hyperbolic transitions, a histogram of shape index values is calculated for each region, and the default threshold values $\varepsilon_{Cone\leftrightarrow Ell} = \pm 0.625$, $\varepsilon_{Cone\leftrightarrow Hyp} = \pm 0.25$ are replaced by the left or right border values of the dominant mode in the shape index histogram, whenever two conditions are satisfied:

1. The mode center must lie in the interval $[0.25, 0.625]$ (for convex cones) or $[-0.625, -0.25]$ (for concave ones) to be properly matched, and

2. Another dominant mode exists left (with positive sign) or right (with negative sign) of it.

As the classification thresholds are automatically computed, the precise values of the shape index are not important. With the following simplistic approximation (5.9), similar results were obtained (figure 5-4 shows a shape classification example):

$$\widetilde{S} = \begin{cases} 1 + 0.5\,(k_{max} - k_{min})/(k_{max} + k_{min}) \in (0.5,\ 1] & if\ k_{max} < 0 \\ -1 + 0.5\,(k_{max} - k_{min})/(k_{max} + k_{min}) \in [-1,\ -0.5) & if\ k_{min} > 0 \\ -0.5\,(k_{max} + k_{min})/(k_{max} - k_{min}) \in [-0.5,\ 0.5] & else \end{cases} \tag{5.9}$$

Using shape indices for classification, the conical shapes and its immediate neighbors (elliptical, hyperbolical) are those eligible for SoR hypothesis generation, but neither the values in the center interval nor the extreme peak or pit values.

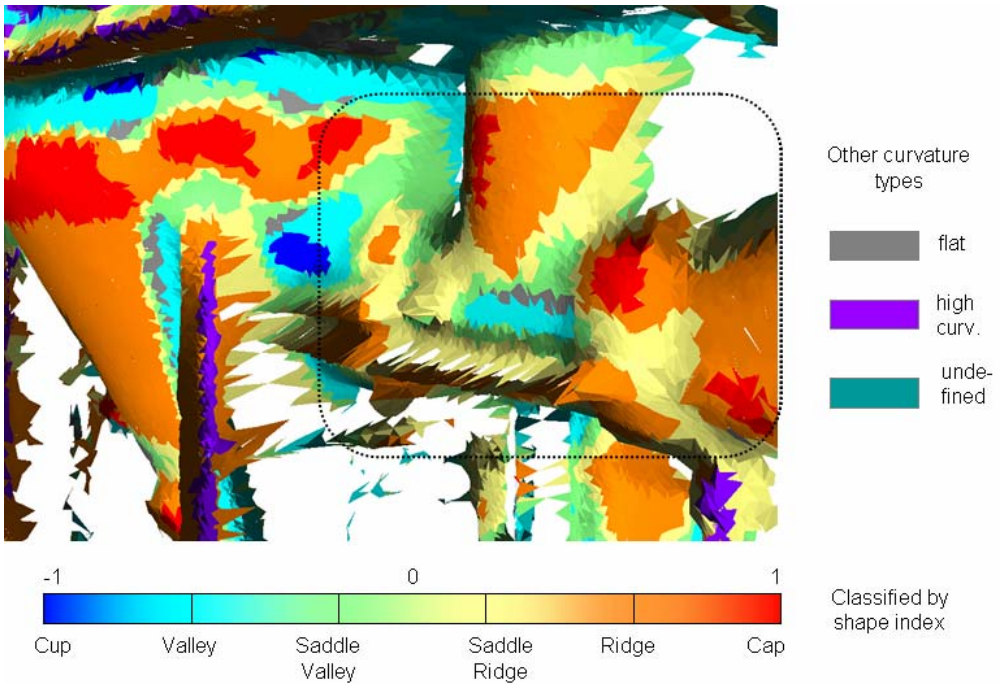

fig. 5-4 Example of local surface classification by shape index; the range image shows part of the THERESA flue gas washer system.

5.1.5 Component labeling using conditional morphology

The next task is to partition the range view into homogeneous connected point sets that represent geometrically distinct components in a process plant: pipe ducts, junctions, flanges, vessels, as well as parts of the surrounding building structure. Ideally, each smoothly curved region corresponds to exactly one SoR hypothesis with linear axis. Algorithms finding optimal partitions, working on *sets* of subsets, may easily attain NP complexity, but a real-time version with *linear* overhead in the image size is desired here.

Unless each range image can be matched and located on-site using an externally provided and up-to-date CAD model, little prior knowledge exists as to the number of components and shape variations to be expected in a random view. The situation differs from medical image analysis: segmenting MRT views from known human organs such as brain or liver tissue is guided by a medical atlas. In plant mapping, a feedback loop from the desired results back to component labeling would be desirable in order to *learn* partitioning rules. This requires for-

malized goodness criteria on the results and an analysis if and how the 'adjusting screws' of an existing algorithm are able to control the results, which is an open research issue.

Empirically, hypothesis generation suffers from two problems caused by partitioning: first, the existence of large and smooth but under-segmented components. For example, a single conical or cylindrical region spilling to an adjacent spherical or planar one via narrow bridges may torpedo reliable estimation of an axis of rotation. Second, small regions from pipe ducts or junctions that are thin and complex-shaped in *relation* to the *spatial* sampling *resolution* provided lead to low confidence in the feature estimation. This reduces the chance to merge adjacent features and properly recover the geometric structure.

Morphological operations on binary and gray-level images, especially *opening* and *closing*, are a fast and proven technique to separate or merge components while largely preserving their size and shape [Ser82][Vin93][SvB99]. They are equally applicable to 3D medical and to 2.5-D range images, viewing the points as pixels or voxels in a 2D or 3D grid defined by the scanning order.

An important issue is explicit control of component topology under morphological operations, regarding the number of components and holes. In thinning and skeletonization applications the goal often is to *preserve* the topology [CSM07][DoB03]. Our motivation is, somewhat contrarily, to *deliberately change* it with *minimum effort*, say, at most three erosions to break a large component into smaller ones, or three dilations to merge adjacent components, and to always know about topology changes. Only by splitting dissimilar parts or merging similar ones will the new partition likely improve the hypothesis generation; nothing may be gained from merely adding or removing some points at the boundary.

Conditional Morphology Approach: The set of regions eligible for SoR hypothesis generation is sorted by size. The largest and the smallest ones are selected for conditional opening and closing, respectively. Conditional opening a region maens applying a bounded number of *conditional erosions* followed by as many *reconstruction dilations*. Conditional closing applies a bounded number of *conditional dilations* followed by as many *reconstruction erosions*. To explain the two main algorithms, the following notations are introduced:

- Connectivity and paths are defined by 8-connectivity, and distances refer to the chessboard, or geodesic, distance. On the regions' complement, 4-connectivity applies therefore [Vin93]. $N_8(p)$ resp. $N_4(p)$ denote the 8- resp. 4-neighborhood of a pixel p, including p.

- Erosion, dilation: the two morphological operations apply to arbitrary pixel sets P seen as binary images under their characteristic functions. The structuring element is a 3x3 mask with all one's, therefore erosion $\varepsilon(P) := \{p \in P : N_8(p) \subseteq P\}$, dilation $\delta(P) := \bigcup_{p \in P} N_8(p)$.

- Homogeneity: the predicate $H_C(p)$ asserts that the surface at p is locally smooth, i.e. p is neither a jump edge nor a crease edge point, has a curvature type eligible for SoR hypothesis generation, and is identical for all pixels assigned to the same region C.

- Component labels, surface assignment: $l(p)$ denotes the number (label) of the component or region (both terms are used synonymously) C that pixel p is assigned to; $l(p)<0$ indicates an undefined component label. The label of a component C is denoted by l_C.

- Geodesic dilation ('flood fill'), CC (connected component) labeling: a region C can be enumerated from any point $p \in C$ (or, generally, from a subset $M \subseteq C$ called 'marker image') by repeated dilations increasing the distance from point p (or set M) by 1. The geodesic distance $d(p, q)$ between two points $p, q \in C$ denotes the length of the shortest path inside C leading from p to q and obeying the connectivity. Formally,

$$C = \bigcup_{j=0}^{\infty} D_j(p), \quad D_j(p) := \{q \in C : H_C(q) \wedge l(q) < 0 \wedge d(p,q) \le j\} \tag{5.10}$$

  Practically, starting from a seed point p, flood-filling a component with label c labels the following sequence of set increments ΔC_j until the increments become empty:

$$C = \dot{\bigcup_{j \ge 0}} \Delta C_j(p),$$
$$\Delta C_j(p) := \begin{cases} \{q \in N_8(\Delta C_{j-1}(p)) - \Delta C_{j-1}(p) : H_C(q) \wedge l(q) < 0\} & \text{if } j > 0 \\ \{p\} & \text{if } j = 0 \wedge H_C(p) \wedge l(q) < 0 \\ \emptyset & \text{else} \end{cases} \tag{5.11}$$

  Labeling an arbitrary pixel set runs once through the set, incrementing the label for each p found with $\Delta C_0(p) \neq \emptyset$ and executing the labeling sequence (5.11). This CC labeling takes always linear running time in $|P|$.

- Crossing numbers [SvB99]: For a point p in a region C, the crossing number $X_5(p, C)$ reports the maximum number of *0-1*-transitions along two circles centered at p with radii one and two. A *0-1*-transition occurs where not the current pixel but its successor belongs to the set C.

  $X_5(p, C) > 0$ holds for any boundary point unless p is a singleton component. If $X_5(p, C) = 1$, removing p from C can neither disconnect C nor change the number of holes in C. A point p with $X_5(p, C) = 1$ is called a *simple point*.

  The measure $X_5(p, C)$ is a form of the Hilditch crossing number (Svensson and Borgefors [SvB99]) designed to detect topology changes from local properties of pixel neighborhoods; it is extended here from 3x3 to 5x5 pixel masks. Deleting all points eroded in one pass *simultaneously* may disconnect the component even if each point alone is classified as simple according to the 3x3-neighborhood test. For example, a two-pixel-wide bridge would be completely removed in a single erosion pass [CSM07].

The following algorithm *ConditionalErode* (Img, C, d, L) processes a 'large' region C referring to a range image Img and is given the current nesting depth d which must stay below some maximum value d_{max}. It returns a list L of result components along with their dilation depths, and is formulated easiest as a recursive algorithm shown on the following page.

Its counterpart, a simple algorithm *ReconstructDilate* not shown here processes the component list L left by *ConditionalErode* cyclically and 'breadth-first' in the sense that each component C_i is dilated *once* if $d_i > 0$, and is removed from the list, otherwise. During dilation, the grown pixels are labeled with their according component number.

Algorithm *ConditionalErode* (**in out** RangeImage Img, **in** Region C, **in** int d,
in out ComponentList L)

Find the boundary set $B := \{p \in C : N_8(p) \not\subset C\} = C - \varepsilon(C)$;
Find the set of non-simple points $BNS := \{p \in B : X_5(p) > 1\}$;
if ($BNS \neq \emptyset$) // possible topology change
{
Delete pixel labeling on the interior $C \backslash B \subset Img$;
Do component labeling on $C \backslash B$: $C \backslash B = C_1 \cup \ldots \cup C_k$, $C_i \cap C_j = \emptyset$ for $i \neq j$;
for $i=1 \ldots k$
if (C_i is *large* $\wedge\ d<d_{max}$)
ConditionalErode (Img, C_i, $d+1$, L);
else
Append (C_i, d) to L;
}
else if ($d<d_{max}$) // no topology change, more erosion possible
{
Delete pixel labeling on boundary $B \subset Img$;
ConditionalErode (Img, $C \backslash B$, $d+1$, L);
}

ConditionalDilate processes a list L of 'small' components needing dilation and returns a list L_E of components to be reconstructed at the end by erosion; L_E is initially empty. The main loop of *ConditionalDilate* has the following pseudo-code notation:

Algorithm *ConditionalDilate* (**in out** RangeImage Img, **in out** ComponentList L,
out ComponentList L_E)

ListPosition $current$ = L.first;
while ($L \neq \emptyset$)
{
Get component (C, d) in list L at position $current$; // d is nesting depth of C
if ($d<d_{max}$)
{
d++;
Dilate $D = \delta(C, M) \subset Img$ returning a set M of component labels;
$D = \{q \in N_8(p) : p \in C \wedge H_C(q) \wedge l(q) < 0\}$
$M = \{l(q) \geq 0 \wedge l(q) \neq l_C : q \in N_8(p)\ for\ some\ p \in D\}$
if ($\exists$ component C': $l_{C'} \in M \wedge$ mergeable(C, C'))
{
Assign label l' to point set $D \subset Img$ and add D to C'; // Merge C to C'
Remove C from L;
if ($\neg C'$ is *small* $\wedge$ (C', d') $\in L$) // C' found in L
Move C' from L to L_E, as (C', max (d, d'));
}
}
else
Move C from L to L_E, as (C, d);
$current$ = L.next; // L is processed cyclically
}

Two components merged in the dilation algorithm must have compatible, i.e. equal or similar curvature types or adjacent types in the shape space. A region C' that has been merged but is no longer considered 'small' (dilatable) is removed from list L.

The counterpart of *ConditionalDilate* is a simple algorithm *ReconstructErode* not shown here; it processes the list L_E cyclically and 'breadth-first', by eroding each component once, decrementing its nesting depth d, and removing it from the list as d becomes *0*. Eroded pixels are unlabelled.

Some examples of breaking up under-'segmented' regions by conditional opening and merging over-'segmented' ones by closing are shown in figures 5-5 to 5-7. In general, experience has shown that conditional opening works reliably and produces well 'workable' regions. The closing operation does not prove equally effective in merging small adjacent regions. Also, closing does not strictly preserve size as it should, for the following reason: if a smooth region is bounded by edge points, e.g. self-occluding jump edges, it cannot grow beyond its borders - this is part of the conditions of dilation. Conversely, the following erosion will erode the smooth region interior, i.e. shrink its size. There is another good reason for dilating, though: it provides a cheap way of calculating pixel distances and inferring region adjacency. These properties may however be calculated by distance transformations without actually dilating the pixel sets.

One general objection against using *intermediate* concepts of region homogeneity and partitioning is that neither curvature types nor morphological operations are truly targeted towards our *goal* where a point set is considered a *homogeneous* region if it is rotationally symmetric with respect to a straight axis of rotation. This basically supports Rabbani's conclusion preferring *under-segmented* decompositions [Rab06] which are easier to refine than conversely regrouping and merging over-segmented ones.

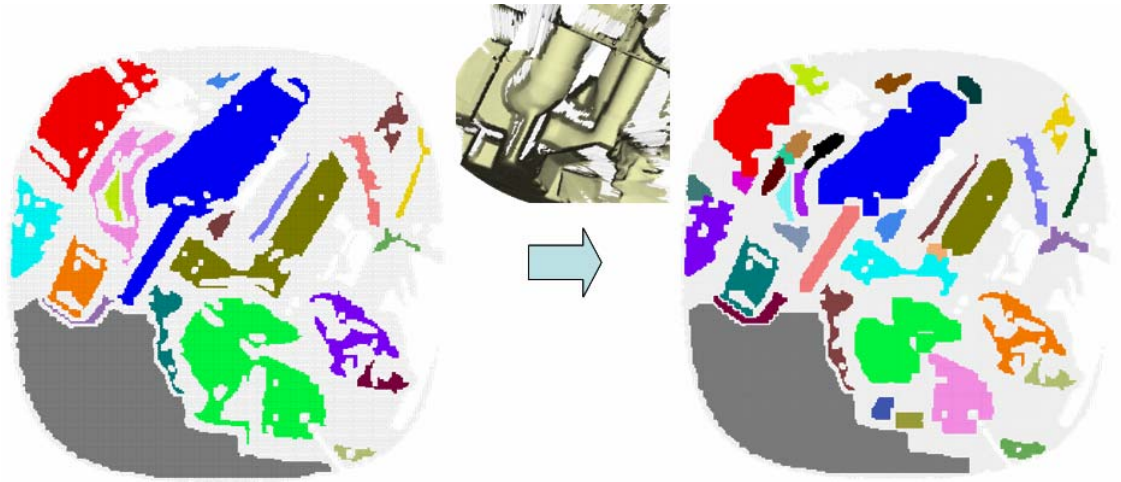

fig. 5-5 Morphology example showing the range view Baugruppe_6201_vw2. Left: initial region labeling based on mean curvature sign and 10 levels of curvedness. Right: regions after conditional opening followed by closing, each 4 pixels wide. Regions are shown here in the *non-metrical* 2D scanning view. For a better visual impression in this and following figures, the small icon on top (centre) shows a 'similar' 3D metrical view of the same image.

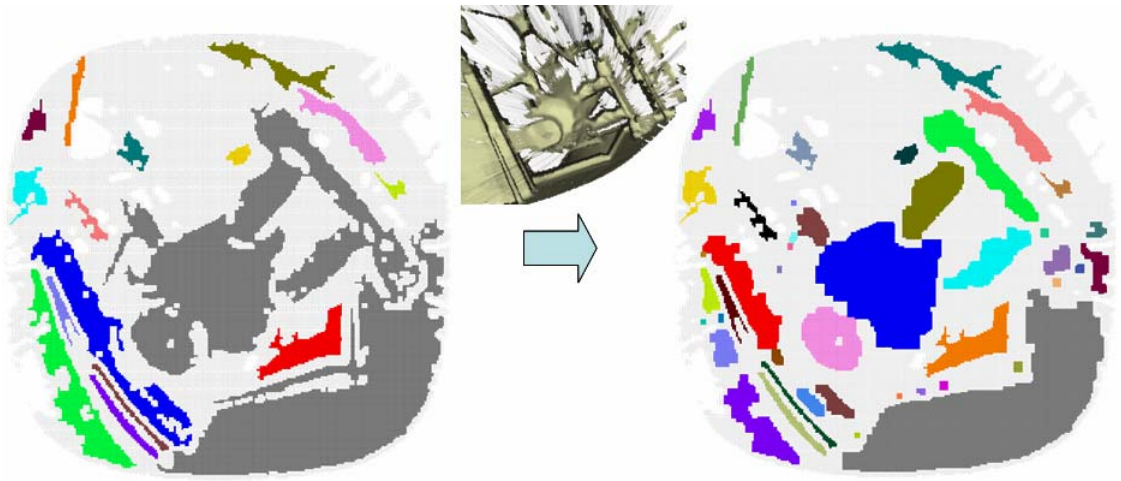

fig. 5-6 Morphology example showing range view Entspanner_2005_vw8. Left: initial region labeling with a large 'under-segmented' gray region. Right: regions after conditional opening.

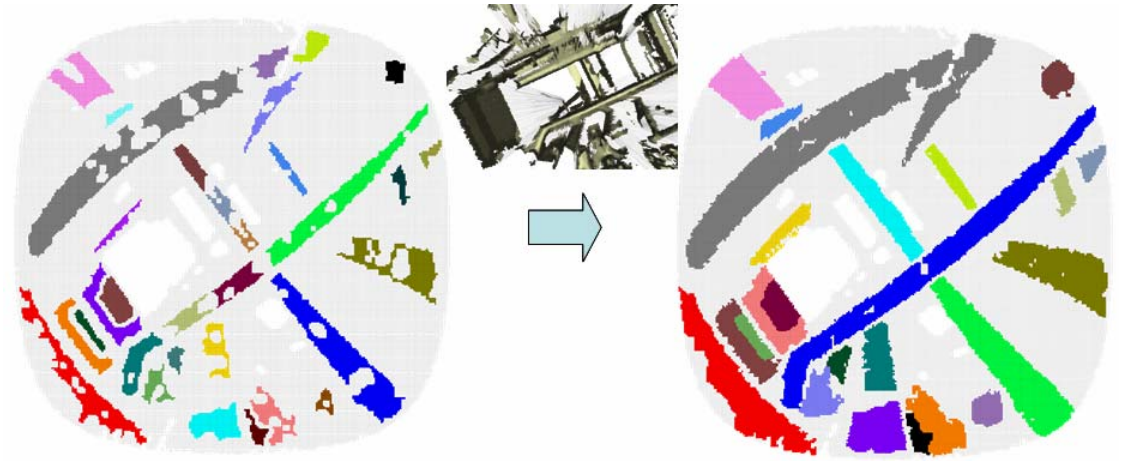

fig. 5-7 Morphology example showing range view Wasserleit_vw3. Left: initial region labeling based on K-/H- signs leading to 'over-segmented' pipes. Right: regions after pure conditional dilation.

5.1.5.1 Basic component features

Component labeling provides access to the region's point set. Simple *properties* of *components*, beyond *point* or *point neighborhood properties* can therefore be calculated. Mainly three features will be frequently used and referred to in the hypothesis generation in sections 5.2-5.4, the *curvature sign* being *convex* or *concave*, a coarse estimate of the curvature radius denoted $r^{(0)}$, and a component 'foot point' denoted $\mathbf{f}^{(0)}$:

$$
\begin{aligned}
&K_{\{min|max\}} := \frac{1}{N}\sum_i \kappa_{\{min|max\},i} \qquad \overline{H} := \frac{1}{N}\sum_i H_i \qquad \Rightarrow \\
&CC\ is \begin{cases} convex & if\ \overline{H} > 0 \\ concave & if\ \overline{H} < 0 \end{cases} && Curvature\quad sign \\
&r^{(0)} \approx \frac{1}{\max\left(|K_{min}|, |K_{max}|\right)} \left(\approx \frac{1}{2|\overline{H}|} \quad if \quad K_{min} \approx 0 \vee K_{max} \approx 0 \right) && Curvature\quad radius \\
&\mathbf{f}^{(0)} := \frac{1}{N}\left(\sum_i \mathbf{p}_i - \frac{1}{2\overline{H}} \cdot \sum_i \mathbf{n}_i \right) && Foot\quad point
\end{aligned}
\tag{5.12}
$$

Index i runs over N points $\mathbf{p}_i$ in the component. The second approximation of the curvature radius using the Mean curvature $H=(\kappa_{min}+\kappa_{max})/2$ is justified when either the average minimum or the average maximum curvature is close to 0, which holds for the regions of interest. The point $\mathbf{f}^{(0)}$ coarsely approximates the region center located on the axis of rotation; it applies to convex and concave surfaces alike.

5.2 Parameter estimation by chain pursuit

5.2.1 Motivation and problem statement

A common intermediate goal towards the estimation of SoR parameters, mainly the axis, is to estimate a *Darboux* coordinate frame [NuM88], characterizing the normal, tangent, and cross section directions of an entire *component* region. At point level, the *principal curvature* directions κ_{min}, κ_{max} in section 5.1.4 give a first clue about tangent and cross section directions, but at an inadequate resolution of the 8-neighborhood.

For *smooth* surfaces, the *Euler formula* $\kappa_\theta = \kappa_{max}\cos^2\theta + \kappa_{min}\sin^2\theta$ indicates an unknown curvature κ_θ in a direction including a known angle θ with the direction of κ_{max}, denoted as a *normal section*. For *sampled data*, the true values of κ_{max}, κ_{min}, and the angle θ are unknown. One may consider (least-squares) solving from known curvatures in four directions of the 8-neighborhood, similarly as proposed by Chen and Schmitt and others [ChS92][HSh03]. However, the *scan lines* defined by the 8-neighborhood do not form valid normal sections which the Euler formula assumes; in general, they are neither tangential to the true surface nor perpendicular to the surface normal approximation used (5.1.1), nor orthogonal to each other. Viewing the computational effort to remedy these problems at point level, it seemed more effective to simply overlook larger neighborhood (window) sizes around each point.

One obvious *efficient* way of getting a better directional resolution is extending point curvature to *path integrated curvature*, taking paths (or chains) of unbounded length from randomly selected seed points. Paths in the *tangent* and in the *cross* section direction will be grown, exploring forward and backward until the region of homogeneous curvature is left or a discontinuity, i.e. a jump or crease edge point, or the image border is encountered.

5.2.2 Forming MIN/MAX chains

Let $P:=(\mathbf{p}, \mathbf{p}_1,\ldots\mathbf{p}_n)$ denote a path of 8-connected points starting at a point $\mathbf{p}=\mathbf{p}_0$. The discrete path curvature at $\mathbf{p}$ along path P is calculated as follows:

$$k(\mathbf{p}, P) := \frac{1}{n}\sum_{i=0}^{n-1} k(\mathbf{p}_i, \mathbf{p}_{i+1}). \tag{5.13}$$

Minimum resp. maximum path curvature values over all paths of length n are sought:

$$\begin{aligned} k_{min}^{(n)}(\mathbf{p}) &:= \min_{P=(\mathbf{p}, \mathbf{p}_1,\ldots,\mathbf{p}_n)} k(\mathbf{p}, P) \\ k_{max}^{(n)}(\mathbf{p}) &:= \max_{P=(\mathbf{p}, \mathbf{p}_1,\ldots,\mathbf{p}_n)} k(\mathbf{p}, P) \end{aligned}. \tag{5.14}$$

The corresponding arguments, the paths minimizing resp. maximizing the curvature sum in (5.14) are called MIN resp. MAX paths or, synonymously, chains.

Geometric interpretation: On a conical surface, one path approximates the *tangent direction* for large *n* and has a path curvature close to 0. It is obtained by the MIN chains on a convex surface, and by the MAX chains on a concave surface, the distinction being made according to the curvature sign in formula (5.12). On a cone with small opening angle, in particular on a cylinder, a path curvature of *maximum magnitude* runs orthogonally to the tangent direction and approximates a circular *cross section* of constant curvature radius r. The maximum magnitude is obtained by MAX chains in case of a convex surface, and by MIN chains in case of a concave surface. On cones with larger or obtuse opening angles (>90°), the chains maximizing the magnitude of path curvature lose a clear geometric meaning; they may form conic sections or, as a result of discrete sampling, even take a helical, non-planar shape. For surfaces of revolution with a complex radius function, neither the MIN nor MAX chain direction may coincide with the tangent or axis direction. Principal curvature paths are only useful for the subclass of cylindrical and slightly conical surfaces.

Implementation: Paths of minimum and maximum curvature are only approximated, i.e. iteratively grown, maintaining for each path a few (currently 2) candidate end points, with additional state information, e.g. the current end point and the direction leading into this point. Only deviations of at most 45° from this direction *and* from the *average* path direction are allowed. Points on a chain must be neither jump nor crease edge points and their curvature type should match the region curvature type. Path growing stops before a point already assigned to another chain of the same type (MIN or MAX) is encountered. Each new chain and the points on it are marked with a unique ID. A point may belong to at most two chains, one MIN and one MAX chain. Paths must have a minimum number of points (currently 7); otherwise they are deleted. A coarse radius of curvature is also estimated from the points on the path, using formula (5.12). See figure 5-8 for an illustration of the MIN/MAX chain pursuit.

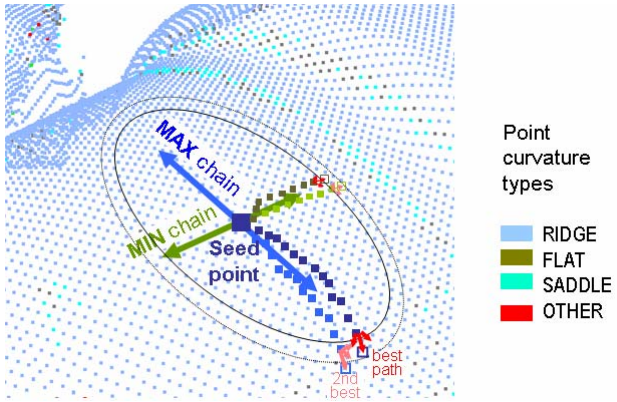

fig. 5-8 MIN and MAX chain pursuit on a range image (isbigwy1 from Michigan State University)

As connected points are assigned to regions, chains are assigned to connected chain components and labeled on-the-fly. Two chains have the same label if they share a point (then one chain must be of MIN, and the other one of MAX type), or if the 8-neighborhood of some point contains a point of the second chain. Each connected component of chains forms a candidate for a circular cone segment. Besides accurate estimation of principal curvature

direction and magnitude, MIN and MAX chains serve to 'coat' the smooth and gross conical structure relevant for fitting with some mesh of chains, but to omit obscuring details such as screw joints, nuts, flanges, cables or sheathings that may be present.

Summarizing, a chain has the following attributes:

- Running identification number
- Chain type (MIN or MAX)
- Region curvature type
- Identification of the connected chain component it belongs to
- Number of points
- Start point $\mathbf{s}$, and direction vector $\mathbf{d}$ from the start to end point (not normalized)
- Curvature radius r
- Mean point normal $\mathbf{n}_c$ (for MIN chains)

For each seed point, the run time complexity of chain pursuit is linear in the number of points visited and bounded by the *diameter* of the region in chessboard distance, which is in the order of the square root of the number of points in the region. The overall complexity, independent of the seed sampling rate, is linear because processed chain points are *marked* and then never used again. In figure 5-9 are shown some examples of chains found and linked on cylindrical and conical surface regions, by randomly sampling from the seed points.

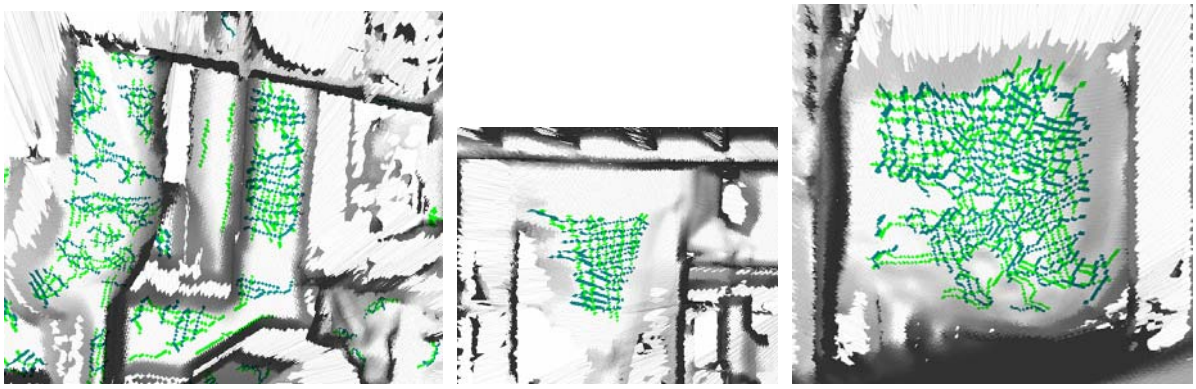

fig. 5-9 MIN chains (shown in light green) and MAX chains (in dark green) extracted on different straight and curved cylinder surfaces (left), on a conical surface (middle), and on an SoR of mixed cylindrical/ellipsoidal type (right).

5.2.3 Initial cone parameters from chains

An algorithm for estimating cone parameters from regions of curvature types RIDGE and VALLEY is proposed using MIN and MAX chains [Lös05]. A parameter may depend on previous estimates of other parameters and undergo several refinements. MIN chains yield coarse tangent directions (no axis direction yet!), and MAX chains give estimates of the radius as a function of position in the tangent direction. These estimates help to reject outlier chains. From the surviving chains and the radius function a set of foot points is estimated through which an axis line is fitted. In case of a truly 'conical' radius function, an apex point can be directly obtained. By projecting the chain points onto the axis, improved estimates of

the radius as well as the interval of axis projection are obtained. At every stage, the cone hypothesis may fail. The data flow is illustrated in figure 5-10 and explained in more detail.

Step 1: Determine a mean tangent unit direction $\mathbf{t}$ from the MIN chain direction vectors $\mathbf{d}_i$; by adding direction vectors of different lengths, they are implicitly weighted. If necessary, flip directions to be consistent, i.e. have a positive dot product with the current mean $\mathbf{t}^{(i)}$. The tangent $\mathbf{t}$ serves as a direction to project the chains onto, i.e. order them by positions.

Step 2: Build a histogram of chain curvature radii, and define bounds (low and high percentiles) for outlier rejection. For each inlier chain, project the chain center point onto the tangent direction $\mathbf{t}$, i.e. compute $s = (\mathbf{s} + 0.5 \cdot \mathbf{d})^T \cdot \mathbf{t}$. Collect pairs of projection values and curvature radii (s_i, r_i) from the valid chains, and estimate the radius function of projection $r(s)$ as a robust least-squares line through them; i.e. repeatedly discard samples with an error greater than a multiple of the root mean square error (rms) until no more outliers remain or until the rms falls below some threshold. Finally, estimate the half opening angle $\delta/2 = \tan^{-1}(r(1) - r(0))$.

Step 3, MIN chain filter: check the MIN chain directions for mutual consistency. If two of them $L_i := L(\mathbf{s}_i, \mathbf{d}_i)$ and $L_j := L(\mathbf{s}_j, \mathbf{d}_j)$ were true cone tangent lines, then, using the geometric relations in figure 5-11a, their included angle θ should depend on the known opening angle δ, the radius $r_i = r(\mathbf{s}_i)$ at point $\mathbf{s}_i$, and the distance $d(\mathbf{s}_i, L_j)$ of the starting point $\mathbf{s}_i$ from the peer tangent line L_j as follows:

$$\theta = \sin^{-1}\left(\frac{d(\mathbf{s}_i, L_j)}{r_i} \cdot \sin\left(\frac{\delta}{2}\right) \right) \tag{5.15}$$

The actually measured included angle θ_{ij} between directions $\mathbf{d}_i$ and $\mathbf{d}_j$ compared to the angle expected (5.15) is a measure of mutual consistency of the MIN chain directions; the chains causing the largest angular deviations are discarded repeatedly.

Step 4: Construct points lying on the axis of rotation, called *foot-points*, and estimate an initial axis from the foot point set $\{\mathbf{f}_i\}$, using their center as the axis position and the direction of largest spread as the axis direction.

The assumption here is that some MAX chains of roughly cross-sectional shape exist; this holds true only for cones with a small opening angle. Points lying on an ideal circular cross section have two properties that allow for finding foot points.

1. The chord (line segment) linking start and end point defines a plane being orthogonal to the chord direction and passing through the chord *bisector*; this plane contains the axis.
2. Normal rays of points on the cross section all intersect the axis in a *single point* (by property (3.5), they intersect the axis, and the cross section is also orthogonal to the axis).

In fact, two simple but slightly different methods based on these properties were implemented. The first one [Lös05] (figure 5-11b) combines facts 1 and 2: intersect the chord plane with those two normal rays passing through the start resp. the end point, and take the mean of the two intersection points as a foot point $\mathbf{f}$. Those rays include the smallest angles $\rho < \pi/2$ with the plane normal; therefore, their intersection position has lowest sensitivity. The

position sensitivity is proportional to $\cos^{-2}(\rho)$ and approaches ∞ as $\rho \to \pi/2$. The second method only uses fact 2: find a single *point* $\mathbf{f}$ minimizing the *mean orthogonal distance* from all normal rays constructed from the points of a MAX chain (closed least-squares solution).

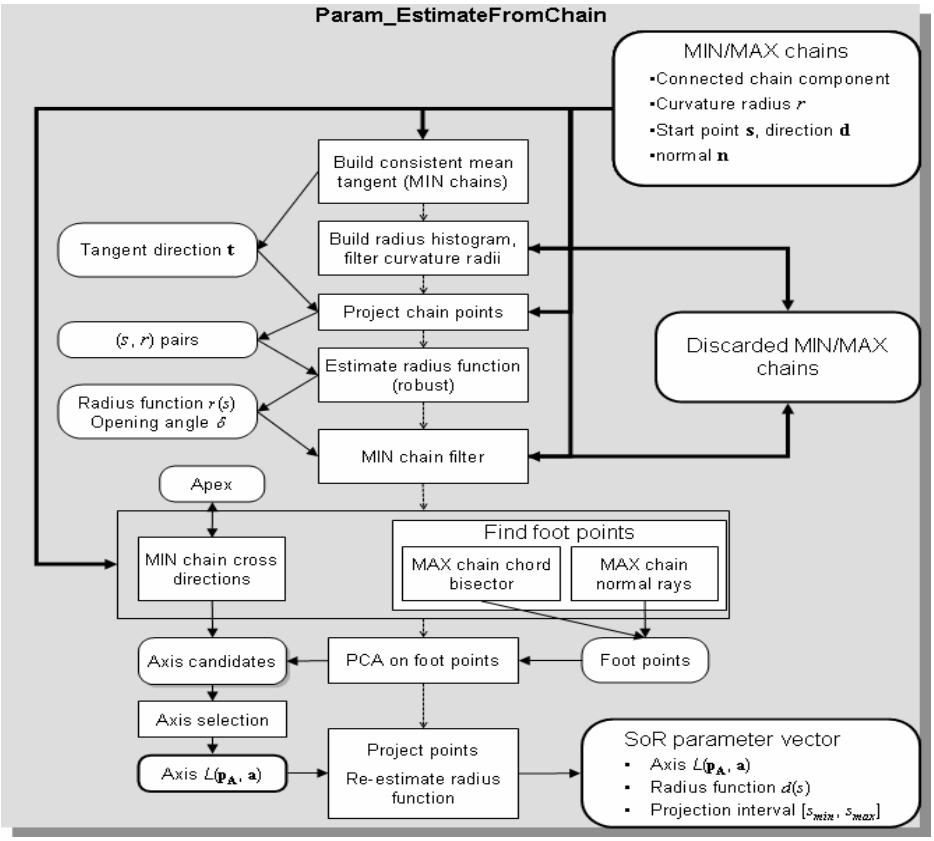

fig. 5-10 Data flow diagram of the CHAIN algorithm for SoR parameter estimation; rounded boxes with bold lines denote external interface data (classes).

Information from the MIN chains can also be used to construct or to refine an axis estimate, assuming that the MIN chains are roughly tangential to the surface:

1. All *tangential rays*, i.e. MIN chains seen as infinite lines $L(\mathbf{s}_i, \mathbf{d}_i)$, approximately meet in a common apex point, unless the surface is a cylinder in which case the apex is a vanishing point. Find a point minimizing the mean orthogonal distance from all tangential rays.
2. For any MIN chain *tangent direction* $\mathbf{t}_c$, the unknown axis direction is some linear combination of $\mathbf{t}_c$ and the *chain normal* $\mathbf{n}_c$ (the mean point normal). In other words, the axis direction is orthogonal to the cross product $\mathbf{q} = \mathbf{n}_c \times \mathbf{t}_c$. Therefore, find a direction which is most orthogonal to all $\mathbf{q}_i$ constructed from MIN chains in this way; it is the eigenvector direction of least spread (smallest eigenvalue) of the set $\{\mathbf{q}_i\}$. As tangent direction $\mathbf{t}_c$, either the MIN chain direction itself or, if an apex point has been found in 1., the vector connecting the chain start point with the apex may be taken.

Many axis points and several candidates of axis directions may be available from the methods in this step (see algorithm in figure 5-10). The candidate axis minimizing the mean orthogonal distance from all normal rays will be selected.

Step 5: Estimate the radius function and the interval of projection $[s_{min}, s_{max}]$ occupied by the chain points, using this time the axis estimated in step 4. Estimating the radius function is described in the following subsection 5.2.4.

The algorithm has several (planned) points of failure where the current hypothesis is abandoned. For example, too few chains or too few valid foot points have been found, or more 'outliers' than 'inliers' in a robust fitting procedure.

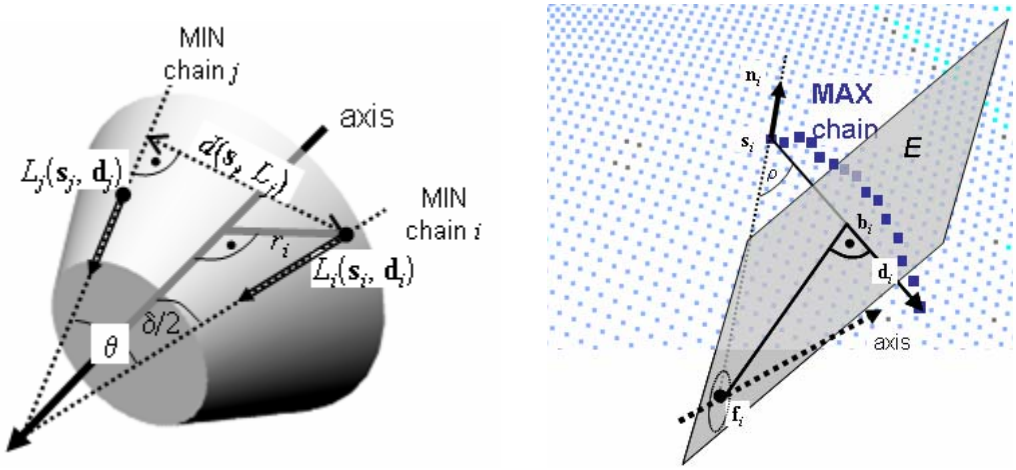

fig. 5-11 a) Geometric relationship between different MIN chain directions (left), b) Plane *E* through a MAX chain bisector and its intersection $\mathbf{f}_i$ with a normal ray $L(\mathbf{s}_i, \mathbf{n}_i)$ (right).

5.2.4 Estimating the radius function

If a SoR with an estimate of its axis line $L(\mathbf{p}_A, \mathbf{a})$ and point samples $\{\mathbf{p}_i\}_{1 \le i \le N}$ on the surface are available, the radius function assuming circular cross sections is estimated as follows:

- Calculate projections $s_i = s(\mathbf{p}_i, L)$ and radii $r_i = d(\mathbf{p}_i, L)$ onto the axis (3.6), and calculate minimum and maximum projection values s_{min}, s_{max}.

- Subdivide the projection interval $[s_{min}, s_{max}]$ and assign the points to bins B_h $(0 \le h < H)$ of equal width. The bin width Δs respectively number of bins H are chosen to have a given *minimum average* number of points in each bin. Alternatively, a statistical formula may be used to choose the bin width [Sco79] if the standard deviation of projection values is known.

- Calculate bin values $s[h]$, $r[h]$: set $s[h]$ to the grid value $s_{min} + h \cdot \Delta s$, and set $r[h]$ to a smoothed average radius from two adjacent bins. Despite ideally circular cross sections, in practice, the radii are often not symmetrically distributed around their means, but there are mainly outliers with *larger* radii. The points causing them are found on surfaces tangential to the SoR, and are often 'mixed points' due to the laser footprint. Instead of the biased average, the median of radius values is chosen as the bin radius:

  $$r[h] = \underset{s_i \in B_{h-1} \cup B_h}{\text{Median}} \; r_i \quad (1 \le h < H).$$

- Find a polygon approximation $r(s)$ of the bin sample values $(s[h], r[h])$. If a regression line returns a rms error that is within the one expected from the sample sizes and the dis-

tance uncertainty, report a cone or cylinder and return the regression line intercept and slope as parameters. Otherwise, repeatedly subdivide the sequence $(s[h], r[h])$ at points with maximum radius error until all sub-sequences have a small error. Report a general SoR radius function, and return the sequence $(s[h_i], r[h_i])$ $(1 \le h_1 < h_2 < \ldots < h_k < H)$ as a piecewise linear approximation.

The algorithm will be referred to as *UpdateProjRadius* and used throughout this work for estimating the SoR radius function once the axis parameters are known.

5.3 Foot point transformations

5.3.1 Basic foot point transformation (FPT)

The SoR ray property (3.5) leads to efficient procedures for estimating unknown axis parameters directly, without constructing chains and even without requiring noisy estimates of principal curvatures and principal directions. In fact, from a few neighboring 3D points presumably part of a single SoR, the according radii and axis *points* ('foot points') may be easily estimated. When this is done systematically on ordered point sets, we use the term *foot point transformation*. The main advantage of these algorithms is their *locality* - being applicable to small point sets, giving valid estimates unaffected by partial occlusion - as well as their simplicity and generality. Axis curve and radius function may have any shape.

If M is a set of sample points $\mathbf{p}_i$ with normals $\mathbf{n}_i$ $(i=1..N)$, the foot point $\mathbf{f}(M)$ is defined as the point with the smallest mean squared distance from all N normal rays $L(\mathbf{p}_i, \mathbf{n}_i)$, i.e.

$$\mathbf{f}(M) = \arg\min_{\mathbf{x}} \frac{1}{N} \sum_{i=1}^{N} d_{\perp}^2(\mathbf{x}, L(\mathbf{p}_i, \mathbf{n}_i)) \tag{5.16}$$

By (3.6), and abbreviating $\Delta\mathbf{p}_i := \mathbf{x} - \mathbf{p}_i$:

$$d_{\perp}^2(\mathbf{x}, L(\mathbf{p}_i, \mathbf{n}_i)) = \Delta\mathbf{p}_i^T \left(\mathbf{I}_3 - \mathbf{n}_i^T \mathbf{n}_i\right)^T \cdot \left(\mathbf{I}_3 - \mathbf{n}_i^T \mathbf{n}_i\right) \Delta\mathbf{p}_i \ .$$

Therefore, (5.16) is rewritten as

$$\mathbf{f}(M) = \arg\min_{\mathbf{x}} \frac{1}{N} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2, \qquad \mathbf{A} := \left[\mathbf{I}_3 - \mathbf{n}_1^T \mathbf{n}_1, \ldots, \mathbf{I}_3 - \mathbf{n}_N^T \mathbf{n}_N\right]^T \in \Re^{3N \times 3},$$
$$\mathbf{b} := \left(\mathbf{p}_1^T \left(\mathbf{I}_3 - \mathbf{n}_1^T \mathbf{n}_1\right), \ldots, \mathbf{p}_N^T \left(\mathbf{I}_3 - \mathbf{n}_N^T \mathbf{n}_N\right)\right)^T \in \Re^{3N} \tag{5.17}$$

If the matrix $\mathbf{A}$ has full rank, the point with minimum distance is therefore found in closed form as a linear least-squares solution:

$$\mathbf{f}(M) = \left(\mathbf{A}^T \mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{b} \tag{5.18}$$

A foot point operator *repeatedly* calculates points according to equations (5.17, 5.18) by *sliding* a *mask* M_{ij} containing m^2 points over a range image, i.e.

$$M_{ij} := \{\mathbf{p}_{kl} : |i-k| \le m/2, |j-l| \le m/2\},\ 1 \le i \le n_{Rows},\ 1 \le j \le n_{Cols}.$$

In the implemented software, updating the system ($\mathbf{A}^T\mathbf{A}\in\Re^{3\times3}$, $\mathbf{A}^T\mathbf{b}\in\Re^3$) in (5.18) is realized by a sliding window operator like the normal estimation explained in section 5.1.1. Adding one new 3D point and removing an old one means to add or subtract one 3x3-matrix $\mathbf{I}^3\text{-}\mathbf{n}\mathbf{n}^T$ and one vector $(\mathbf{I}^3\text{-}\mathbf{n}\mathbf{n}^T)\mathbf{p}$ each time. The advantage is keeping the computation cost to a minimum, largely independent of the mask size m. Also, the foot point operator is only applied to regions having the appropriate curvature type, and possibly to only those ones corresponding to regions in previous images where SoR were already detected.

On a planar or gently curved patch, the normals become linearly dependent and the solution point (5.18) diverges as the matrix $\mathbf{A}^T\mathbf{A}$ becomes singular. A similar behavior is observed on saddle-type surfaces, especially of MINIMAL type where both principal curvatures have the same magnitude but opposite signs. On ridge or valley shaped patches, the foot points tend to concentrate around the axis curve as intended. Likewise, near peaks (domes) or pits (troughs), foot points concentrate around the center points or around a short center curve segment.

The FPT concentrates a fairly evenly sampled spatial distribution of surface points around thin clusters approximating curved axes of rotational symmetry. Superficially and in purpose, it resembles a Hough transform which accumulates (concentrates) model parameters estimated from points in an abstract parameter space, the Hough space. However, a Hough transform estimates *complete* objects, i.e. *global* properties from local data, thereby leveraging noise and complexity (each data point generates many object candidates), whereas FPT generates only one axis point and radius for a data point, but no object hypothesis. Further steps are needed to analyze the point clusters and to estimate the object parameters from them (section 5.3.4).

The main drawback of the basic FPT is its *sensitivity*: the foot point location is sensitive to small *normal variations* because it minimizes orthogonal distance from normal rays without restrictions. This is especially true when the system matrix is almost singular.

5.3.2 Constrained foot point transformation (C-FPT, DC-FPT)

To reduce the sensitivity, the distance of the foot points from the *surface points* will be constrained. We consider the family of point sets $M(r) = \{\mathbf{p}_i(r) := \mathbf{p}_i + r\,\mathbf{n}_i\}_{i=1..N}$, depending on a scalar r. All points $\mathbf{p}_i(r)$ have the same distance $|r|$ from their surface point $\mathbf{p}_i$. Assuming that the set $M=M(0)$ is a circular cross section, the centers (sample means) $\mathbf{cp}(r)$ of the sets $M(r)$ will 'converge' to the desired axis point for *some* r which is the value *minimizing the sample variance* (spread) of the $\mathbf{p}_i(r)$ within their sets $M(r)$. With the notations

$$\begin{aligned} \mathbf{cp} &:= \frac{1}{N}\sum \mathbf{p}_i, \quad \Delta\mathbf{p}_i := \mathbf{p}_i - \mathbf{cp} \\ \mathbf{cn} &:= \frac{1}{N}\sum \mathbf{n}_i, \quad \Delta\mathbf{n}_i := \mathbf{n}_i - \mathbf{cn}, \qquad \mathbf{c}(r) := \frac{1}{N}\sum \mathbf{p}_i(r) = \mathbf{cp} + r\,\mathbf{cn} \end{aligned} \tag{5.19}$$

The sample variance $Var\,(r)$ of points in the set $M(r)$ is written as a function of r:

$$Var(r) = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{p}_i + r\mathbf{n}_i - \mathbf{c}(r))^T(\mathbf{p}_i + r\mathbf{n}_i - \mathbf{c}(r)) = \frac{1}{N-1}\sum_{i=1}^{N}(\Delta\mathbf{p}_i + r\Delta\mathbf{n}_i)^T(\Delta\mathbf{p}_i + r\Delta\mathbf{n}_i) = \frac{1}{N-1}\left(\sum_{i=1}^{N}\Delta\mathbf{p}_i{}^T\Delta\mathbf{p}_i + 2r\sum_{i=1}^{N}\Delta\mathbf{p}_i{}^T\Delta\mathbf{n}_i + r^2\sum_{i=1}^{N}\Delta\mathbf{n}_i{}^T\Delta\mathbf{n}_i\right) \quad (5.20)$$

By setting $\partial Var(r)/\partial r = 0$, the variance minimizing radius and the associated foot point follow:

$$r_{\text{var min}} = -\left(\sum_{i=1}^{N}\Delta\mathbf{p}_i{}^T\Delta\mathbf{n}_i\right)\Big/\left(\sum_{i=1}^{N}\Delta\mathbf{n}_i{}^T\Delta\mathbf{n}_i\right), \quad \mathbf{f}(M) = \mathbf{c}(r_{\text{var min}}) \quad (5.21)$$

Only a minimum of variance is possible since $\partial^2 Var(r)/\partial r^2 = \sum_i \Delta\mathbf{n}_i{}^T\Delta\mathbf{n}_i \geq 0$.

Again, implementation by incremental update and down-date works as efficiently as for the basic FPT (5.18), or even more so. The reader may notice (5.21) having a similar form as the reciprocal directional curvature for a point pair (5.6). The estimate (5.21) can be applied in two ways:

Single-stage, direction-independent: A window mask M is slid over the entire image or the components of interest, paying no regard to the, as yet unknown, *directions* of rotational symmetry. If the point set M is not a circular cross section to any specific direction, the estimates r_{varmin} may be meaningless, at least inaccurate. The following sub-section 5.3.3 attempts to overcome this drawback by combining radius and direction estimation.

Two-stage, directional: A suitable direction $\mathbf{t}$ is *chosen* for each component, to ensure that point sets be narrow, approximately circular cross sections with respect to $\mathbf{t}$. How to find it? Any tangent direction of small curvature magnitude, similarly to a MIN chain in section 5.2, will suffice. More effective methods will be discussed in the next section 5.4. Next, all points are projected onto direction $\mathbf{t}$ and the projection interval divided into narrow bins ensuring a minimum bin population. In the second stage, C-FPT (5.21) is applied to each bin B_h ($1 \leq h \leq H$), and a foot point $\boldsymbol{f}_h$ is obtained. In the sequel, the two-stage procedure will be referred to as *directional c*onstrained foot point transformation (DC-FPT).

5.3.3 Isotropic constrained foot point transformation (IC-FPT)

If a component region is conjectured to lie on a SoR with approximately linear radius function, i.e. a cone, this hypothesis can be tested, and initial axis and radius parameters be found, without knowing a direction to project the points onto.

A *mean radius* r_0 is sought first so as to minimize the variance of $M(r) = \{\mathbf{p}_i(r) := \mathbf{p}_i + r\,\mathbf{n}_i\}_{i=1..N}$. This time, an *entire region* is used as point set. In other words, the goal is to select r_0 for which the set $M(r)$ becomes as thin as possible. This holds when the scatter matrix $\mathbf{S}(r)$ achieves a maximum ratio between its largest eigenvalue and the two smaller ones:

$$r_0 = \arg\max_r \frac{EW_{max}(\mathbf{S}(r))}{EW_{mid}(\mathbf{S}(r))} \quad where \quad \mathbf{S}(r) = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}_i(r) - \mathbf{c}(r))^T(\mathbf{p}_i(r) - \mathbf{c}(r)) \quad (5.22)$$

For computation, the 2nd-degree matrix polynomial $\mathbf{S}(r)$ is written as a sum of constant coefficient matrices $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{N}$ - compare (5.20) - independent of r:

$$\begin{aligned}\mathbf{S}(r) = \quad & \mathbf{P} + r\mathbf{Q} + r^2\mathbf{N} \quad \textit{where} \\ & \mathbf{P} = \frac{1}{N}\sum_i \Delta\mathbf{p}_i^T \Delta\mathbf{p}_i, \quad \mathbf{Q} = \frac{2}{N}\sum_i \Delta\mathbf{p}_i^T \Delta\mathbf{n}_i, \quad \mathbf{N} = \frac{1}{N}\sum_i \Delta\mathbf{n}_i^T \Delta\mathbf{n}_i \in \Re^{3,3}\end{aligned} \tag{5.23}$$

The eigenvalues of $\mathbf{S}(r)$ in (5.23) - zeroes of the characteristic polynomial of matrix $\mathbf{S}(r)$ - are polynomials of degree six in r, and their ratio a rational function of degree six which may be minimized numerically or in discrete steps Δr. The *eigenvector* to the largest eigenvalue EW_{max} gives an approximate axis direction $\mathbf{t}_0$ as well, and the minimizing r_0 yields an axis center point $\mathbf{cp} + r_0\,\mathbf{cn}$. The radius r_0 should be *negative* for convex, and *positive* for concave cones.

In a second step, the *slope* t of the radius function depending linearly on the projection to $\mathbf{t}_0$ is found again as a variance minimizing value of the foot point cloud $M(r, t)$ which now depends on two parameters, r and t:

$$M(r,t) = \{\mathbf{p}_i(r, t) := \mathbf{p}_i + r\,\mathbf{n}_i + t\,(\mathbf{p}_i^{\mathrm{T}}\mathbf{t}_0)\,\mathbf{n}_i\}_{i=1..N}.$$

Similar to (5.23), the scatter matrix $\mathbf{S}(r, t)$, a bivariate polynomial matrix of degree two in r and t, is rewritten as a sum of constant coefficient matrices:

$$\begin{aligned}\mathbf{S}(r,t) = \; & \mathbf{P} + r\mathbf{Q} + r^2\mathbf{N} + t\mathbf{U} + t^2\mathbf{V} + rt\mathbf{W} \quad \textit{where} \\ & \mathbf{U} = \frac{1}{N}\sum_i \Delta\mathbf{p}_i^T \Delta\tilde{\mathbf{n}}_i, \quad \mathbf{V} = \frac{1}{N}\sum_i \Delta\tilde{\mathbf{n}}_i^T \Delta\tilde{\mathbf{n}}_i, \quad \mathbf{W} = \frac{1}{N}\sum_i \Delta\mathbf{n}_i^T \Delta\tilde{\mathbf{n}}_i \in \Re^{3,3} \\ & \tilde{\mathbf{n}}_i := \left(\mathbf{p}_i^T \mathbf{t}_0\right)\cdot \mathbf{n}_i\end{aligned} \tag{5.24}$$

To reduce the overhead, minimization of (5.24) with respect to t is done for few values of r only, around the optimum value r_0 (5.22). The matrix expression for which the eigenvalue ratio is maximized effectively integrates the two separate stages of binning the points by projecting and minimizing the spread within each bin. It will be referred to as the *isotropic* or *integrated constrained* foot point transformation (IC-FPT).

For testing the method, a search interval for the radius is defined as follows: first, estimate a coarse radius of curvature from n points using formula (5.12). Then set the search interval to cover some multiple $s>1$ of r_c: $[-s\cdot r_c, -r_c/s]$ if the SoR expected is convex and $[r_c/s, s\cdot r_c,]$ if concave, and divide it into steps Δr; tests were performed with 30 steps. The slope t=tan δ is restricted to the range [-1, 1] and divided into steps Δt=0.1. The ratio between maximum and mid eigenvalue $ewr(r, t)$ obtained for the matrix polynomial $\mathbf{S}(r, t)$ (5.24) is plotted in figure 5-12 for several SoR-like regions setting $s=3$. A lower threshold of 3 was imposed on the eigenvalue ratio ewr to accept a straight axis line and linear radius hypothesis (cone). On rounded crease edges or chamfers of small curvature radius, the eigenvalue ratio appears most sharply peaked.

The results indicate that IC-FPT often provides a coarse but useful and fast estimate of r_0 in (5.22), but not equally reliable estimates of the radius slope t in (5.24). Frequently, t is overestimated on noisy cylinders or acute-angled cones and under-estimated on cones with lar-

ger opening angles, the reason not being known. To estimate the opening angle or the axis direction, in general, the directional constrained algorithm DC-FPT from section 5.3.2 is preferred once the required estimate of the tangent direction is available.

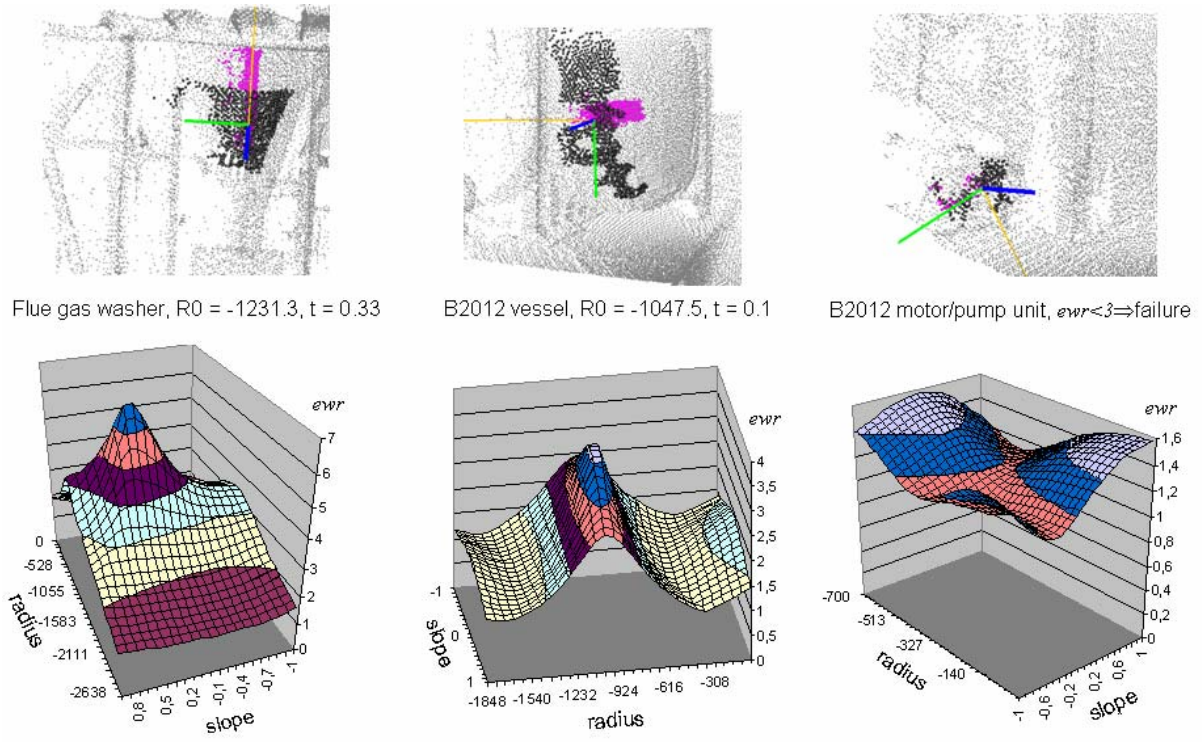

fig. 5-12 Estimating radius (r) and slope (t) in the isotropic foot point transformation (IC-FPT) by maximizing an eigenvalue ratio. Top row: point clouds with selected region points (bold black dots), foot points (magenta), principal frame (yellow-green-blue). Bottom row: Plot of the eigenvalue ratio ewr (r, t) to be maximized.

5.3.4 Axis approximation by principal curves

In the preceding subsections, several algorithms were proposed for the foot point transformation of an entire range view or selected regions from it. The images of rotationally symmetric surfaces generate foot point clouds (FPC) approximating the axis curve. To generate SoR hypotheses, this axis curve needs to be recovered. On SoR segments with linear axis and linear radius function, data points tend to be highly condensed into corresponding FPC segments, whereas near sharp pipe bends or changes in the radius function, or on planar or curved but not rotationally symmetric surfaces the FPC points are scattered or diverging.

Since a FPC is *centered* in the axis, at first glance, one is tempted to seek something like its *medial axis* in 3D. Algorithms from *computational geometry* (CG) calculate medial axes or medial scaffolds [LeK07] as generalizations or related data structures like Voronoi diagrams. Those methods conceive the point set as the *boundary* of a *closed subspace* (a manifold) separating *outer* from *inner* points. While dealing with *unorganized* points they often expect a nearly uniform sampling density. An FPC, however, is a point set of *zero diameter* and zero volume, no manifold, and does not even come close to uniform sampling density. In fact, the FPT will rather destroy any uniformity present in the raw data.

Conceiving an FPC as a noisy one-dimensional structure and processing it by *statistical data reduction* techniques makes more sense. In particular, we look at the concept of *principal curves* [HaS89][KeK00][StR00]. Principal curves generalize the Principal Component Analysis (PCA) on point sets where, in the 3D case, the eigenvector to the largest eigenvalue provides the *direction* of an approximating *line* passing through the cloud *center*. The other two orthogonal eigenvectors indicate directions of deviation or spread. If the points are noisy samples from a line with normally distributed line distances, PCA in fact gives the best *line* approximation, minimizing the squared Euclidean (orthogonal) point distances from the line.

The generalization to samples from *nonlinear* or mixed multi-linear point sets is the principal curve, a smooth (infinitely differentiable) function $\mathbf{f}(t):\Re\rightarrow\Re^d$ of the scalar arc length t which *i*) does not intersect itself, *ii*) has bounded length inside bounded subsets of $\Re^d$, and *iii*) is **self-consistent**, i.e. the curve point $\mathbf{f}(t)$ lies at the expected spatial position of all points that project onto its arc position t:

$$\mathbf{f}(t) = E\left(\mathbf{x} \,|\, s(\mathbf{x}) = t\right) \quad \forall t \tag{5.25}$$

$s(\cdot)$[5] denotes the *projection operator* returning for any point $\mathbf{x}$ the supremum (maximum) arc position t among all points $\mathbf{f}(t)$ on the curve having the closest distance d ($\mathbf{x}$, $\mathbf{f}$) from point $\mathbf{x}$:

$$s(\mathbf{x}) = \sup\left\{t : \left\|\mathbf{x} - \mathbf{f}(t)\right\| = d(\mathbf{x}, \mathbf{f})\right\} \quad \textit{where} \quad d(\mathbf{x}, \mathbf{f}) := \inf\left\{\left\|\mathbf{x} - \mathbf{f}(\tau)\right\| : \tau \in \Re\right\} \tag{5.26}$$

For a straight line $\mathbf{f}$, $s(\cdot)$ is the orthogonal projection and $d(\cdot, \mathbf{f})$ the orthogonal distance.

Kégl et al. [KeK00] not only derive theoretical foundations of principal curves but also develop an algorithm to reduce a point distribution to a polygonal line by iterative refinement. A single line passing through the centre in the direction of the first principal component gives the start values. In the iteration loop one new vertex is added to the current polygon, choosing the midpoint of the longest segment. *All* vertex positions are updated thereafter in an inner loop, alternating between a projection and an optimization step. The projection step partitions the point set according to which polygon segment each point projects. In the optimization step one vertex is moved at a time so as to minimize the geometric distances of all points assigned to its two adjacent segments. The minimized cost function includes further length and angle (curvature) constraints. Kégl constructs a heuristic stopping condition producing roughly $n^{1/3}$ segments. Thereby, the total computational complexity becomes $O(n^{5/3})$.

For our application, a simplistic polygonal approximation is constructed where each new vertex position is set only once. No length or angle constraint as in [KeK00][WoC08] but the magnitude of *point spread* orthogonal to the current direction guides the refinement. Our approximation uses histogram data, i.e. bins of projection, and may produce several disjoint polygons, as long as the projection of a point onto such a structure by (5.26) is well defined.

First, the crucial self-consistency property (*iii*) needs to be adapted to finite sample sets $\mathbf{F} = \{\mathbf{x}_i\}$. Informally, (iii) states that the principal curve is *everywhere well-centered* with respect to

[5] In the original work [HaS89] $t_f()$ denotes the projection.

the point set: the curve value $\mathbf{f}(t)$ at any arc position t is the expectation, respectively, the sample *mean* of all points that project onto t. In case of a *finite* point set $\mathbf{F}$ projecting onto a real interval, the set of these points is empty almost everywhere. To assign a function value to any t, points $\mathbf{x}$ with *nearby projections* should also contribute to it to a degree that exponentially decays with their projection distance $|t - s(\mathbf{x})|$ from t:

$$\mathbf{f}(t) = \sum_{\mathbf{x}\in\mathbf{F}} e^{-(t-s(\mathbf{x}))^2} \cdot \mathbf{x} \Big/ \sum_{\mathbf{x}\in\mathbf{F}} e^{-(t-s(\mathbf{x}))^2} \tag{5.27}$$

Every $\mathbf{x}$ in (5.27) contributes to all positions, but, in practice, distant points will be dropped. In our application (5.27) is replaced by a much simpler expression: the projection space itself is quantized into bins or buckets of fixed width Δ_s, and inside each bin all points count equally.

$$\mathbf{f}_h := \mathbf{f}(t_h) = \frac{1}{|\mathbf{F}_h|} \sum_{\mathbf{x}\in\mathbf{F}_{hi}} \mathbf{x}, \quad where \qquad t_h = \left(h + \frac{1}{2}\right) \cdot \Delta_s, h = 0,1,\ldots,$$
$$\mathbf{F}_h := \left\{\mathbf{x} \,\middle|\, s(\mathbf{x}) \in \left[h \cdot \Delta_s, (h+1) \cdot \Delta_s\right]\right\} \tag{5.28}$$

Between the control points t_h, the curve $\mathbf{f}(t)$ is simply linearly interpolated. Before taking the mean (5.28), outlier points frequently arising around locally saddle-shaped or flat structures must be excluded. In fact, the bin medians being less sensitive to outliers will be taken. Also, projection bins may become empty or under-populated; a minimum bin population $N_h \geq 10$ is required. As the bin centers $\mathbf{f}_h$ in (5.28) become undefined for certain grid values t_h, they will be interpolated using the closest neighbor bins with well-defined centers.

How should the projection interval $[s_{min}, s_{max}]$ be subdivided? A paper by Scott [Sco79] derives a formula for the optimal histogram bin width Δs minimizing the expected squared frequency error with respect to the continuous density function approximated by the histogram. This integral error shows a trade-off between quantization error or *bias* which increases with the bin width, and *variance* of histogram values which decreases as the expected sample size per bin, or bin width, increases. In case of N (=$|\mathbf{F}|$) normally distributed projection samples[6] with standard deviation σ_s the bin width and, accordingly, the number of bins H and the average population size per bin N_h should be:

$$\Delta_s \approx 3.49 \cdot \sigma_s \Big/ \sqrt[3]{N} \quad \Rightarrow \quad H = \left\lceil (s_{max} - s_{min}) / \Delta_s \right\rceil, \quad N_h = N / H \tag{5.29}$$

Suppose the principal curve maps a projection interval $[s_{min}, s_{max}]$ to a single line segment $L(\mathbf{p}, \mathbf{a})$ with point $\mathbf{p}$ and direction $\mathbf{a}$. Using (5.28) to construct a self-consistent curve would require all bin centers $\mathbf{f}_h$ to lie on this line, i.e. $d_\perp(\mathbf{f}_h, L) = 0$ for all h. In case of real, noisy data, this will not hold. To judge the validity of the line approximation, therefore, the bin center distances from the line are related to the mean *overall spread* σ of the foot point cloud. A measure of overall spread is provided by the magnitude of the smaller two PCA eigenvalues $\lambda_{1/2}$ of the FPC, where the largest one, λ_3, corresponds to the main direction $\mathbf{a}$.

[6] The projection lengths are not necessarily normally distributed; the true distribution depends on the shape and on the sampling density. However, Scott's formula is being applied for many distributions.

If not a line but rather a winding space curve, a 3D polygon, or a tree-like edge structure best explains the data, the overall spread would strongly overestimate the mean deviation of points from their as yet unknown best approximant. A more faithful measure is the *local spread* of points in a neighborhood, so to say the *local thickness* of the FPC independent of the shape of the global approximant. An approximate local spread can be obtained by performing PCA inside each bin h, yielding eigenvalues $\lambda_{i,h}$ and corresponding eigenvector directions $\mathbf{e}_{i,h}$ (i = $1 \ldots d=3$). Specifically the portion $\sigma_{\mathbf{a}\perp}[h]$ of the spread orthogonal to (independent of) the *line direction* (**a**) is of interest:

$$\sigma_{\mathbf{a}\perp}[h] = \sqrt{\frac{1}{N_h}\sum_{i=1}^{d}\lambda_{i,h}\cdot\sin^2\left(\angle\left(\mathbf{e}_{i,h},\mathbf{a}\right)\right)} \qquad (5.30)$$

A necessary condition for accepting a line segment $L(\mathbf{p}, \mathbf{a})$ as an approximation is that all bin center distances from the line stay below the local spread within the respective bin, and that the local spread $\sigma_{\mathbf{a}\perp}[h]$ itself stays within the global spread σ, see figure 5-13 for illustration:

$$d_\perp(\mathbf{f}_h, L) < \sigma_{\mathbf{a}\perp}[h] < \sigma \quad (1 \le h \le H) \qquad \text{Relaxed self-consistency} \qquad (5.31)$$

There is no need to reject the linear approximation only because the orthogonal spread σ as such is large, i.e. the point cloud is noisy. If condition (5.31) is violated, an attempt is made to split the current line segment $L(\mathbf{p}, \mathbf{a})$, using some bin center $\mathbf{f}_h$ as an intermediate point. According to the bin numbers participating, denote by $L^{[0,H-1]}$ the original line segment, and by $L^{[0,h]}$, $L^{[h,H-1]}$, respectively, the two split segments. We choose a splitting bin h_{spl} minimizing the weighted squared distance errors of the bin centers from their respective line segments:

$$h_{spl} = \underset{h:0<h<H-1}{\operatorname{argmin}}\left(\sum_{i=0}^{h} w_i \cdot d_\perp^2\left(\mathbf{f}_i, L^{[0,h]}\right) + \sum_{j=h}^{H-1} w_j \cdot d_\perp^2\left(\mathbf{f}_j, L^{[h,H-1]}\right)\right) \qquad (5.32)$$

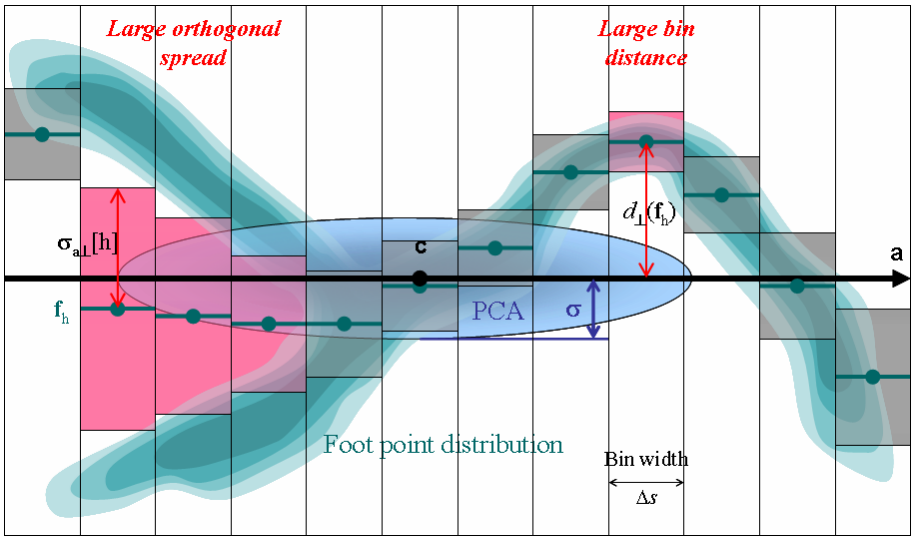

fig. 5-13 A linear (PCA) representation $L(\mathbf{c}, \mathbf{a})$ of a point distribution needs to be refined if the local orthogonal spread $\sigma_{\mathbf{a}\perp}[h]$ at some bin h is large compared to the mean global spread (σ) or if the local approximant $\mathbf{f}_h$ has a large distance from the line $L(\mathbf{c}, \mathbf{a})$ compared to the local spread ($\sigma_{\mathbf{a}\perp}[h]$).

The splitting bin h_{spl} is accepted only if it leads to a better approximation than the original line $L^{[0,H-1]}$, i.e. achieves a smaller mean squared distance (5.32) than a single line segment. The weights w are normalized bin sizes summing up to 1. After splitting the *projection step* follows: the points are re-distributed, i.e. assigned to the two line segments $L^{[0,h]}$ and $L^{[h,H-1]}$:

- Each bin $i<h$, in its entirety, is assigned to line segment $L^{[0,h]}$, and each bin $i>h$ to $L^{[h,H-1]}$.
- Each point $\mathbf{x}$ in bin h is assigned to the line segment it projects onto, if it projects onto exactly one. It is assigned to the closer line segment (only) in case that it projects onto both or onto none of them. The *closer* of two segments is the one with the smaller orthogonal distance of $\mathbf{x}$ onto $L_{1|2}$ if $\mathbf{x}$ projects onto the segment with $s_{min}<s<s_{max}$, and is the distance to the closer of two endpoints at s_{min} resp. s_{max}, otherwise. This definition is consistent with requirement (5.26) and consistent also with the shortest distance in (3.9).

The segment distribution is updated according to the new point assignment[7] resulting in new line segments: the temporary segments $L^{[0,h]}$ and $L^{[h,H-1]}$ are replaced by the final ones, $L(\mathbf{p}_0,\mathbf{a}_0)$ and $L(\mathbf{p}_1,\mathbf{a}_1)$ obtained from the main PCA directions $\mathbf{a}_{0|1}$ and centers $\mathbf{p}_{0|1}$ calculated for both sub-segments. The line segments will not be contiguous. If prior knowledge tells that contiguity of the principal curve is more important than faithfulness, the temporary line segments $L^{[0,h]}$ and $L^{[h,H-1]}$ connected via bin center $\mathbf{f}_h$ may be preserved. This second approximation strategy always yields a contiguous polygon part of which may however be explained by few data points. A variant of the second strategy connects the two leftmost and rightmost center points whose bins exceed the minimum bin size.

A line segment which cannot be refined is a *terminal segment*. The bin centers may pass the relaxed self-consistency test (5.31) despite a large spread; also, too few points may be left for splitting, or splitting does not improve the criterion value (5.32). None of these cases implies that the line faithfully fits the foot point data. But introducing more line segments, and thereby increasing the curve length, is unlikely to improve the situation. All terminal segments are therefore evaluated by a cost function C composed of a *thinness* term T and a *point density* term D

$$T=\frac{\sqrt{\lambda_1\cdot\lambda_2}}{\lambda_3}\qquad D=\frac{\lambda_1\cdot\lambda_2\cdot\left(s_{max}-s_{min}\right)}{N}\qquad C=w_{thin}\cdot T+\left(1-w_{thin}\right)\cdot D \tag{5.33}$$

here $\lambda_{1|2|3}$ denote the PCA eigenvalues in the order of increasing magnitude, s_{max} and s_{min} maximum and minimum projection values, N the number of point samples, and w_{thin} a relative weighting factor for thinness and density. The terminal segments are inserted into a list sorted by increasing cost C, where, after normalization of the cost, a small fraction of the total cost will be maintained, only.

The described technique is summarized by the algorithm *PrincipalFtPtCurve* using a list of point addresses (PointList Pts) as input and producing a list of terminal segments (Terminal-

[7] Incrementally, adding or subtracting entire *bin*-PCA's for bins $i\neq h$, and *points* from bin h, according to the above distribution algorithm.

SegmentList TSL). The main data class PointCluster ($\mathbf{p}$, $\mathbf{a}$, s_{min}, s_{max}, Pts) captures a subset of data points with additional line and projection attributes and offers a recursive method *Analyze*() for splitting a segment.

Algorithm *PrincipalFtPtCurve* (**in** PointList Pts, **out** TerminalSegmentList TSL)

Calculate line $L(\mathbf{p}_0, \mathbf{a}_0)$ from Pts by PCA;
Project Pts onto L; calculate s_{min}, s_{max};

PointCluster($\mathbf{p}_0$, $\mathbf{a}_0$, s_{min}, s_{max}, Pts).*Analyze* (TSL);

Calculate normalized costs and retain lowest-cost segments from TSL;
return TSL;

```
PointCluster::Analyze (TerminalSegmentList TSL)
{
   Partition [s_min, s_max] into H bins;                          //Using (5.29)
   for p in Pts
      Assign s(p) to bin h(p); Update histogram bin variables (f_h, Pts_h) and global spread σ;
   for h<H
      Calculate distance d(f_h, L(p, a)), local orthogonal spread σ_a⊥[h];
                                                                  // (5.30)
   if d(f_h, L) > σ_a⊥[h] ∨ σ_a⊥[h] > σ for some h                //Consistency criterion (5.31)
      CalcOptimumSplittingBin (h_spl);                           //Minimize distance error (5.32)

   if splitting bin h_spl exists
   {
      Partition Pts = Pts_L ∪· Pts_R,  Pts_L ⊇ ⋃_{h<h_spl} Pts_h,  Pts_R ⊇ ⋃_{h>h_spl} Pts_h ;
      Calculate new lines L(p_L|R, a_L|R) from Pts_L|R
         using PCA or f_hspl directly;                           //Expectation step
      Project Pts_L|R onto L(p_L|R, a_L|R);                      //Projection step
      PointCluster(p_L, v_L, s_min,L, s_max,L, Pts_L).Analyze (TSL);   //Recursive calls
      PointCluster(p_R, v_R, s_min,R, s_max,R, Pts_R).Analyze (TSL);
   }
   else
   {
      Calculate Cost by (5.33);
      TSL.Insert (TerminalSegment (*this, Cost));                //Insert new terminal segment
   }
}
```

A decomposition result of a foot point cloud by algorithm *Analyze* is shown in figure 5-14.

Computational Complexity

The computational complexity of the algorithm is bounded by $O(N^{4/3})$. The main loop of the algorithm, performed inside the recursive method *Analyze*(), is $O(N)$ because all data points are projected to the line direction and binned. As to the number of recursive calls (line splitting), it has expected $O(log(N))$, but worst case $O(N^{1/3})$ and not $O(N)$ because it is bounded by the number of terminal segments which in turn is at most N_h, the number of bins. From the choice (5.29), $N_h \sim N^{1/3} \cdot (s_{max}\text{-}s_{min})/\sigma_s$; the standard deviation σ_s of point projections onto a ran-

dom direction is a *constant* and has no implicit dependency on N. This gives a total complexity of $O(N^{4/3})$. The expected running time in practice is bounded by the geometric structure complexity and its desired resolution. This is a scene dependent *constant* independent of the number of points N; therefore, the computation cost in practice is $O(N)$. Also, the linear part of the main loop contains only trivial operations; the optimization (5.32) ranging over h takes $O(N^{1/3})$, too. This makes the algorithm amenable to sequential and real-time mapping.

In the theoretical work [KeK00] the factor $N^{1/3}$ plays an important role, too, in the convergence rate and the complexity. We have not yet investigated this relationship.

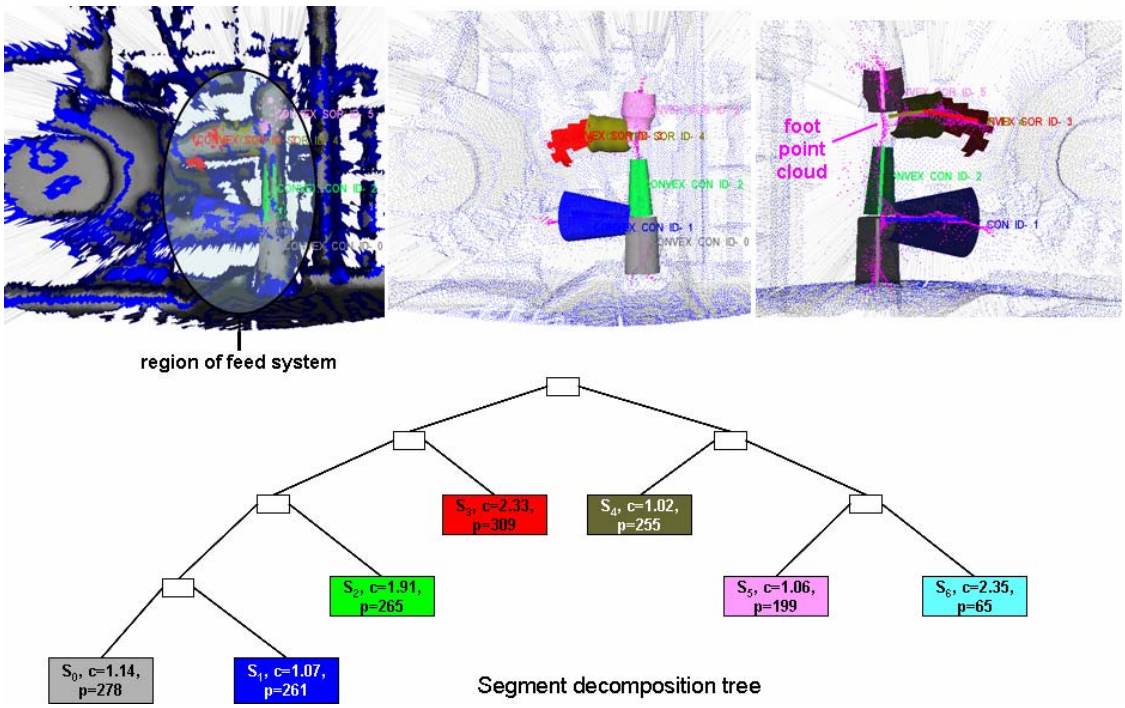

fig. 5-14 Principal Curve algorithm approximating the axis of the feed pipe system of a pressure reducer unit (range view Entspanner_B2005_vw6). The tree symbolizes the linear axis segments produced as leaf nodes with cost c (5.33) and number of points p projecting onto the segment.

5.4 Direct methods for SoR axis estimation

5.4.1 An initial principal component frame

While labeling the seed regions appropriate for SoR detection, basic component properties such as the curvature sign and radius and some 'foot point' $\mathbf{f}^{(0)}$ on the axis are recorded (section 5.1.5.1). A goal remaining is to characterize the principal directions of the surface patch. This is done using principal component analysis (PCA) on the set of *point normal vectors* $\mathbf{n}_i$. The normal's covariance matrix has three orthogonal unit eigenvectors indicating, in the order of increasing eigenvalues, a *tangent* direction $\mathbf{t}$ of minimum curvature, a *cross section* direction $\mathbf{q}$ of maximum curvature, and a *component normal* direction $\mathbf{n}$. These three directions form a coordinate frame (a so-called Darboux frame) centred in the axis point $\mathbf{f}^{(0)}$:

$$\mathbf{F}^{(0)} = \left[\mathbf{f}^{(0)}, \mathbf{R}^{(0)}\right] \qquad \mathbf{R}^{(0)} = \left[\mathbf{t} \quad \mathbf{q} \quad \mathbf{n}\right] \in \Re^{3\times 3}. \tag{5.34}$$

A comment about analyzing the *normal vectors* is in order. Obviously, PCA on the *point samples* would not do the job because the point covariance matrix and its eigenvectors characterize the shape of the *visible* patch but not the underlying 3D surface geometry in an intrinsic, view-independent way. As to the normals, their spread is clearly smaller in the direction of minimum and larger in the direction of maximum curvature magnitude. Both directions are properties of the surface itself. But what is the ranking of the third eigenvalue and the role of its eigenvector? A geometric interpretation - plotting the normals on a *unit Gaussian sphere* - provides little insight. The centre of this normal vector cluster projected onto the unit sphere indicates a *mean sample direction*, but how does this relate to the PCA directions of the normal covariance matrix?

Previous work using range image normals and PCA for statistical shape classification is due to Liang et al. [LTd90][LiT94] and notably Garland [Gar99] who addresses the simplification of polygonal surface meshes and develops a quadric error metric to guide the simplification. He shows [Gar99] that around any point on a quadric surface a *region exists* where the *integral quadric matrix* - the continuous analogue of the discrete normal covariance matrix - has two smaller eigenvalues being proportional to the squared principal curvatures at the reference point. Their corresponding eigenvectors point to the *principal (curvature) directions* whereas the eigenvector to the third and *largest* eigenvalue matches the *surface normal direction*. This result answers the above question, assuming a sufficiently small or a homogeneous region as to curvature type, and justifies our use of PCA on sample normals in order to define the principal directions of a region. It is to note that the component normal eigenvector $\mathbf{n}$ does *not* even up to its sign equal the above mentioned *mean sample normal*.

By [Gar99] the normal PCA matrix approximates a principal frame only locally, but in many practical examples including large though smooth and homogeneous patches the PCA directions do provide a reasonable approximation of the entire region geometry (figure 5-15). Quite often, $\mathbf{t}$ comes close within 5° to the true tangent which in case of cylinders is also the axis direction, The accuracy is then sufficient for a local optimizer to converge to the desired optimum. Figure 5-16 presents several cases where the PCA directions fail to give a useful tangent direction. The method should be used with caution mainly because the initial region labeling does in no way guarantee homogeneous components.

5.4.2 Estimating the cone tangent and axis direction

The following sub-section presents a method for estimating a sufficiently accurate tangent direction which applies to SoR hypotheses with *straight axis* and *linear radius* function, i.e. to truncated *cone* segments. By searching in an angular grid, this method offers a limited resolution of directions, but it has shown to better tolerate badly segmented regions.

Viewing the angular scanner resolution and the low computational effort to be put into primal partitioning, smaller and more distant objects will be affected by severe quantization noise and will produce components with few points of not truly homogeneous curvature type. Figure 5-16 shows a typical example, a 'blurred' conical region spreading across parts of a pump, its motor block and socket. By visual appearance, the PCA tangent direction obtained in (5.34) deviates from the rotational axis by about 50° which a local optimizer cannot compensate for. The problem remains to generate better starting values.

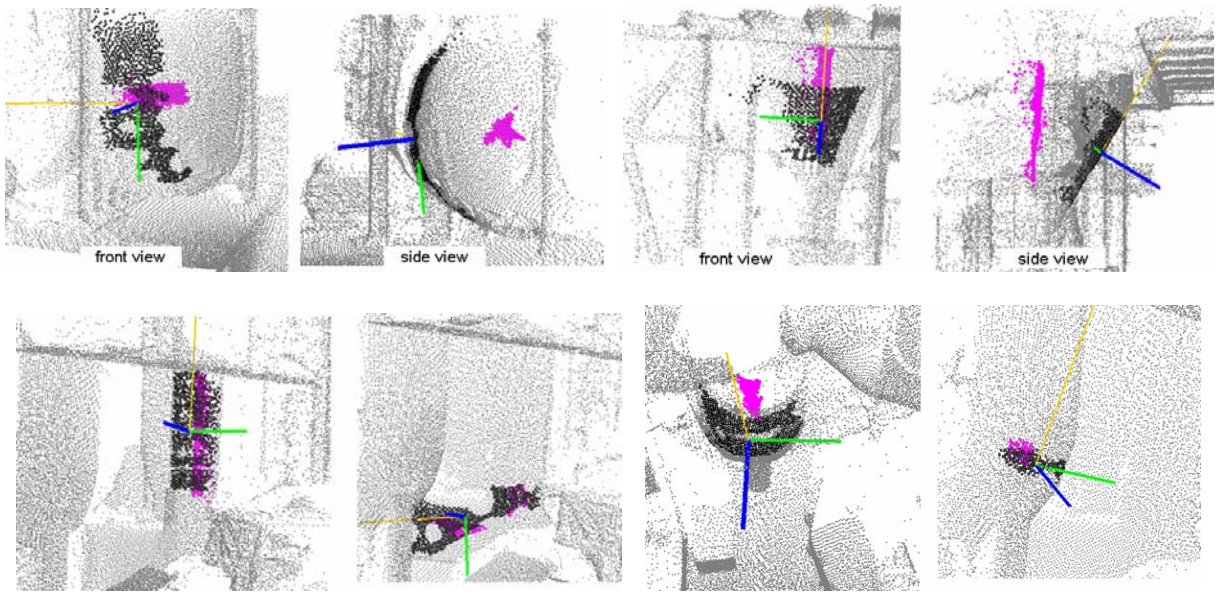

fig. 5-15 Examples of sufficiently accurate principal frame estimation (top left range view: Kessel2012_vw16, top right: RauchgasWäscher_Vw1_slow1, bottom: various SoR regions from Baugruppe_6201). The principal directions (tangent **t**, cross section **q**, normal **n**) are indicated by yellow, green, and blue colors, respectively. Points from the selected component region are highlighted by bold black dots. Magenta dots indicate foot points.

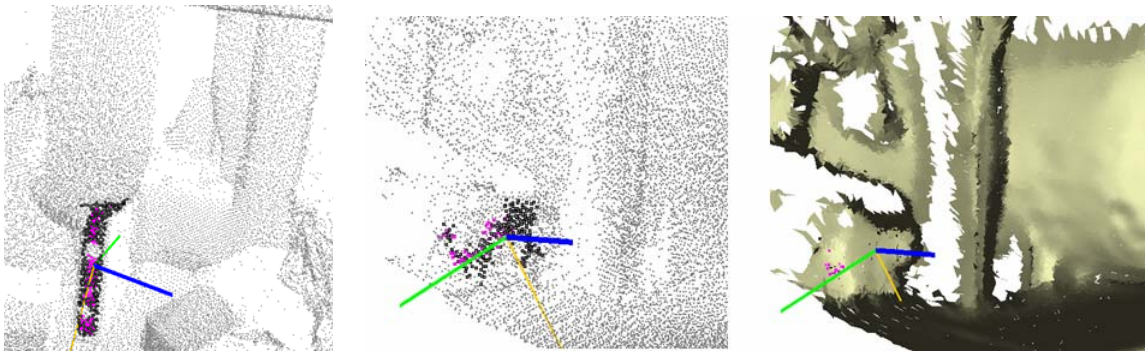

fig. 5-16 Examples of wrong principal frame estimation (left: outlet region from Baugruppe_6201 with inaccurate normal direction, middle and right: pump/motor block region from Kessel_2012 with tangent rotated ≈55° wrongly within the (**t**, **q**) plane.

One obvious idea is to let the tangent vector **t** rotate about the normal vector **n** and choose the rotation angle α_{max} for which the distribution of point distances from the line $L(\mathbf{f}^{(0)}, \mathbf{t})$ becomes *most linear*. To each candidate line $\mathbf{t}(\alpha)$ and N sample points $\{\mathbf{p}_i\}$, projection and distance pairs (s_i, r_i) are calculated, a 2D regression line $r(s) = r_0 + r_1 \cdot s$ is fitted to the (s_i, r_i) samples and the direction is chosen for which the regression error becomes minimal. Alternatively, the normalized *correlation coefficient* R^2 can be maximized; a value of 1 indicates a perfect line fit:

$$\mathbf{t} = \mathbf{t}(\alpha_{max}) \quad where \quad \alpha_{max} = \arg\max_{\alpha} R^2(\alpha), \quad R^2(\alpha) = \frac{\left(\sum_i s_i r_i - N \cdot \bar{s}\,\bar{r}\right)^2}{\left(\sum_i s_i^2 - N \cdot \bar{s}^2\right) \cdot \left(\sum_i r_i^2 - N \cdot \bar{r}^2\right)} \in [0,1]$$

$$\mathbf{t}(\alpha) = \cos\alpha \cdot \mathbf{t} + \sin\alpha \cdot \mathbf{q} \qquad \bar{s} = \sum_{i=1}^{N} s_i, \quad s_i(\alpha) = \Delta\mathbf{p}_i^T \cdot \mathbf{t}(\alpha), \quad \Delta\mathbf{p}_i = \mathbf{p}_i - \mathbf{f}^{(0)} \qquad (5.35)$$

$$\bar{r} = \sum_{i=1}^{N} r_i, \quad r_i(\alpha) = \sqrt{\Delta\mathbf{p}_i^T \Delta\mathbf{p}_i - s_i^2(\alpha)}$$

The nonlinear and differentiable goal function (5.35) may be maximized numerically or directly by discrete search. However, using *mean* values makes it sensitive to *outlier radii* which are observed in particular in the problem cases to be addressed here: radii of points on smoothly joining neighbor surfaces *exceeding* the according cross section radius. Figure 5-17 plots the (s, r) pairs for different tangent rotation angles α, indicating the angle for which the radius function becomes most linear, but it also shows the high number of outliers that are observed on poorly segmented seed regions like the pump-and-motor block in range view Kessel_2012 (bottom of fig. 5-17).

Taking cross section radius *medians* instead of *means* counteracts the bias as long as there are less than 50% outliers. Cross sections are approximated by narrow bins of projection B_h=[$h \cdot \Delta s$, $(h+1) \cdot \Delta s$] as explained in section 5.2.4 on radius function estimation. The bin values ($s[h]$, $r[h]$) are taken as the 2D samples of a linear radius function $r(s)$, and the direction $\mathbf{t}$ yielding the minimum variance of the scatter matrix is returned as the tangent direction. This algorithm is summarized by the following pseudo-code; a result is shown in figure 5-18.

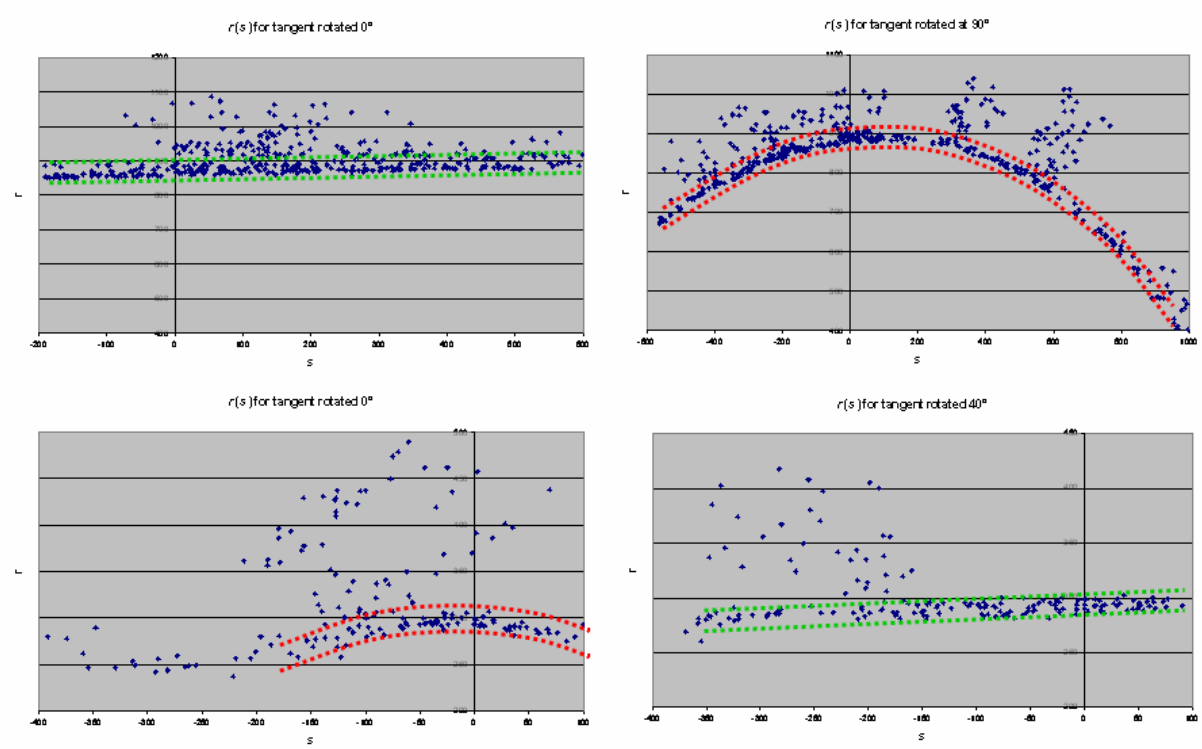

fig. 5-17 $r(s)$ plots for range view Kessel_2012 (top: the tank, bottom: the pump-and-motor block) shown for different tangent angles α. The variance band of inliers for the most linear relationship is shown in green.

Algorithm Vec3d *FindInitialTangent* (**in** PointNormalSet Pts $\{(\mathbf{p}_i, \mathbf{n}_i)\}$, **in** Vec3d $\mathbf{f}^{(0)}$)

Perform PCA on the set $\{\mathbf{n}_i\} \rightarrow$ component directions $\mathbf{t}_0$, $\mathbf{q}$, $\mathbf{n}$;
Vec3d $\mathbf{t} = \mathbf{t}_0$;
for k = 1 …$\lceil \pi/\Delta\alpha \rceil$ // $\Delta\alpha$: angular resolution of direction, algorithm parameter
{
  Project Pts onto line $L(\mathbf{f}^{(0)}, \mathbf{t})$; calculate s_{min}, s_{max};
  for p in Pts
    Assign p to bin B_h with $h=\lfloor (s(p)-s_{min})/\Delta s \rfloor < H$;
  for $h<H$
    Extract projection and median radius values $(s[h], r[h])$ from B_h;
  Perform PCA on the set $\{(s[h], r[h])\} \rightarrow$ eigenvalues λ_{min}, λ_{max};
  if the ratio $\lambda_{min}/\lambda_{max}$ is minimal, set $\mathbf{t}_{min} = \mathbf{t}$;
  $\mathbf{t} = Rot(\mathbf{t}, \mathbf{n}, \Delta\alpha)$; // Rotate $\mathbf{t}$ about $\mathbf{n}$ at angle $\Delta\alpha$
}
return $\mathbf{t}_{min}$;

The computational effort is trivially linear in the number of point samples N because the angular resolution is limited to a constant value ($\Delta\alpha \approx 5\text{-}10°$); locating the median inside a bin costs $O(\log n)$ in the expected number n of points *per bin* which is constant, independent of N.

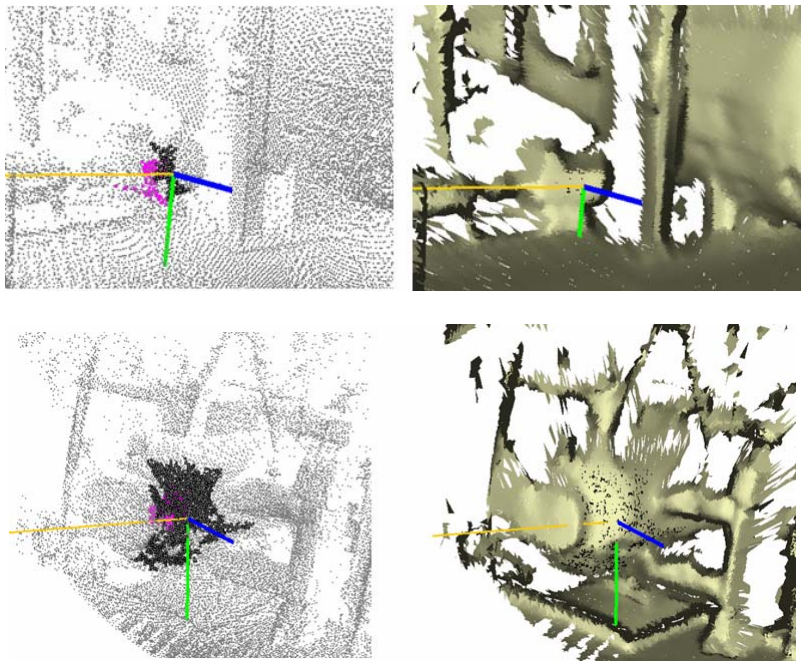

fig. 5-18 Principal frames estimated by algorithm *FindInitialTangent*; top: pump-and-motor block region from Kessel_2012, bottom: region of the pressure reducing unit from Entspanner_2005_vw8.

5.4.2.1 Simplistic axis estimation

The axis direction of a *cone* is a unit vector $\mathbf{a}$ including an unknown *constant* angle with all sample normal vectors, in other words: a direction minimizing the *variance* among angles $\angle(\mathbf{a}, \mathbf{n}_i)$ that the axis includes with all sample normals. Instead of nonlinear angle and unit vector constraints, we may consider *angle cosines*, i.e. dot products $\cos(\angle(\mathbf{a}, \mathbf{n}_i)) = \mathbf{a}^T \cdot \mathbf{n}_i$, giving a linear relationship in the coefficients of $\mathbf{a}$. For small values $s = \mathbf{a}^T \cdot \mathbf{n}_i << 1$, i.e. *small opening angles*, an approximate variance of angle cosines is minimized using the coarse first-order approximation $\cos^{-1}(s) \approx \pi/2 - s$:

$$E_{\approx}(\mathbf{a}) = \sum_i \left(\mathbf{n}_i^T \mathbf{a} - 1/N \sum_j \mathbf{n}_i^T \mathbf{a} \right)^2 = \sum_i \left(\Delta \mathbf{n}_i^T \mathbf{a} \right)^2 \quad where: \quad \Delta \mathbf{n}_i = \mathbf{n}_i - \mathbf{cn} \,. \tag{5.36}$$

Next, the axis direction $\mathbf{a}$ is confined to the *plane* spanned by the normal direction $\mathbf{n}$ already known from PCA on the component normals (section 5.4.1), and the tangent direction $\mathbf{t}$ estimated by the *FindInitialTangent* algorithm. Therefore, the unit vector $\mathbf{a}$ is expressed as a linear combination of $\mathbf{t}$ and $\mathbf{n}$, leaving a rotation angle α as the only free parameter:

$$\mathbf{a}(\alpha) = \mathbf{t} \cos\alpha + \mathbf{n} \sin\alpha \,. \tag{5.37}$$

The error function (5.36) as a function of α reads:

$$E_{\approx}(\alpha) = \sum_i \left(\left(\Delta \mathbf{n}_i^T \mathbf{t} \right) \cos\alpha + \left(\Delta \mathbf{n}_i^T \mathbf{n} \right) \sin\alpha \right)^2 \quad where: \quad \Delta \mathbf{n}_i = \mathbf{n}_i - \mathbf{cn} \,. \tag{5.38}$$

Setting its derivative to zero obtains the following closed-form solution for α:

$$\frac{\partial E_{\approx}(\alpha)}{\partial \alpha} = 0 \quad \Rightarrow \quad \alpha = \frac{1}{2} \tan^{-1} \left(\frac{2 \sum_i \left(\Delta \mathbf{n}_i^T \mathbf{n} \right) \left(\Delta \mathbf{n}_i^T \mathbf{t} \right)}{\sum_i \left(\Delta \mathbf{n}_i^T \mathbf{t} \right)^2 - \sum_i \left(\Delta \mathbf{n}_i^T \mathbf{n} \right)^2} \right) \tag{5.39}$$

Application of equation (5.39) to real range images from the THERESA plant did however not result in many useful direction estimates. Apparently, even for cones with small opening angles or for cylinders, the first order approximation is used far outside its domain of validity for many sample normals. Using a more accurate third order approximation $\cos^{-1}(s) \approx \pi/2 - s - s^3/6$, on the other hand, yields no closed solution (5.39). Therefore, other techniques are needed than minimizing angular variance.

5.4.3 Axis estimation by minimizing ray distances

In this sub-section, the characterizing normal ray property for SoR (3.5) will be used to estimate the axis direction. The axis should be the line achieving in the mean sense smallest orthogonal squared distances from the sample normal rays $L(\mathbf{p}_i, \mathbf{n}_i)$. An initial axis point guess $\mathbf{f}^{(0)}$ is known (5.12); let $\mathbf{x}$ denote the unknown axis direction. With the connection vectors $\Delta\mathbf{p}_i := \mathbf{p}_i - \mathbf{f}^{(0)}$ between ray $L(\mathbf{p}_i, \mathbf{n}_i)$ and axis $L(\mathbf{f}^{(0)}, \mathbf{x})$, their shortest distance becomes

$$d_{L\perp}(\Delta\mathbf{p}_i,\mathbf{n}_i,\mathbf{x})=\begin{cases}\Delta\mathbf{p}_i{}^T\cdot\dfrac{\mathbf{n}_i\times\mathbf{x}}{\|\mathbf{n}_i\times\mathbf{x}\|} & \text{if } \mathbf{n}_i,\mathbf{x} \text{ linearly independent}\\ \|\Delta\mathbf{p}_i\|\cdot sin\angle(\Delta\mathbf{p}_i,\mathbf{x}) & \text{else}\end{cases} \tag{5.40}$$

The axis direction $\mathbf{x}$ is found by minimizing

$$\mathbf{a}=\underset{\mathbf{x}:\|\mathbf{x}\|=1}{\arg\min}\ E(\mathbf{x}),\quad E(\mathbf{x}):=\sum_{i=1}^{N}d_{L\perp}^2\ (\Delta\mathbf{p}_i,\mathbf{n}_i,\mathbf{x}) \tag{5.41}$$

Ignoring the degenerate linearly dependent case in (5.40) the error function (5.41) may be rewritten as follows, making clearer its bilinear dependence on the unknown $\mathbf{x}$:

$$E(\mathbf{x})=\sum_{i=1}^{N}\frac{\mathbf{x}^T\cdot\overbrace{(\Delta\mathbf{p}_i\times\mathbf{n}_i)(\Delta\mathbf{p}_i\times\mathbf{n}_i)^T}^{\mathbf{A}_i}\cdot\mathbf{x}}{\mathbf{x}^T\cdot\underbrace{(\mathbf{I}_3-\mathbf{n}_i\mathbf{n}_i^T)}_{\mathbf{B}_i}\cdot\mathbf{x}} \tag{5.42}$$

Analyzing the error function (5.42) which is a sum of Rayleigh quotients, one might conjecture a generalized eigenvalue/eigenvector solution, at least for each matrix term ($\mathbf{A}_i$, $\mathbf{B}_i$) individually. Yet, no closed-form solution is found minimizing the sum. Nor did directly expanding the gradient vector $\nabla E(\mathbf{x})$ yield a closed solution.

5.4.4 Axis estimation using algebraic line geometry

Finding a line minimizing orthogonal distances from N given lines becomes tedious and leads to nonlinear equations if a solution is attempted directly in 3D. Surprisingly, the equations become *simpler*, namely linear in the unknowns and solvable in closed form in $O(N)$ time, by *embedding* the problem in a *higher-dimensional* vector space. Constraints between the geometric structures become simpler due to the partially redundant structure representation. This approach is due to Pottmann et al. [HOP05] [OPW05] and has been used to detect and describe symmetries of more general surfaces than SoR, namely those described by *equiform motions* of points in space.

A 3D line, such as a normal ray $L(\mathbf{p}, \mathbf{n})$ going through a surface point $\mathbf{p}$ in the direction of the point normal $\mathbf{n}$, is uniquely determined by three parameters: two polar angles describing the unit direction $\mathbf{n}$, and the orthogonal line distance from the origin. This line is as well described by *two* direction vectors: $\mathbf{n}$ and the direction $\mathbf{p}\times\mathbf{n}$ of the plane spanned by $\mathbf{n}$ and the vector $\mathbf{p}$ connecting the line to the origin (assuming $\mathbf{p}\neq\mathbf{0}$):

$$(\mathbf{p},\mathbf{n})\ \rightarrow\ (\mathbf{n},\mathbf{p}\times\mathbf{n})=:(\mathbf{n},\overline{\mathbf{n}})\in\Re^6 \tag{5.43}$$

The two orthogonal 3D vectors ($\mathbf{n}$, $\mathbf{p}\times\mathbf{n}$) called (6D) *Plücker coordinates* do not depend on the particular point because any valid point $\mathbf{p}'$ on the line satisfies $\mathbf{p}'=\mathbf{p}+\lambda\mathbf{n}$, therefore $\mathbf{p}\times\mathbf{n}=\mathbf{p}'\times\mathbf{n}$. N lines L_i ($i=1,\ldots,N$) intersect a further line L if and only if their Plücker coordinates $(\mathbf{n}_i,\overline{\mathbf{n}}_i)$ all satisfy the *linear* equations

$$\bar{\mathbf{n}}_i^T \cdot \mathbf{x} \;+\; \mathbf{n}_i^T \cdot \bar{\mathbf{x}} \;= 0 \quad (i = 1,\ldots,N) \qquad\qquad \left(\mathbf{x} \neq \mathbf{0} \in \Re^3, \bar{\mathbf{x}} \in \Re^3, \mathbf{x}^T \bar{\mathbf{x}} = 0\right) \tag{5.44}$$

for some non-zero solution $(\mathbf{x}, \bar{\mathbf{x}})$ vector that is the Plücker representation of the axis line L (see fig. 5-19 for illustration). To see this, N lines $L_i(\mathbf{p}_i, \mathbf{n}_i)$ intersecting a single line denoted $L(\mathbf{q}, \mathbf{x})$, holds if and only if all orthogonal line distances vanish. This is equivalent to

$$(\mathbf{n}_i \times \mathbf{x})^T (\mathbf{p}_i - \mathbf{q}) = 0 \quad \Leftrightarrow \quad \underbrace{(\mathbf{p}_i \times \mathbf{n}_i)}_{\bar{\mathbf{n}}_i}{}^T \mathbf{x} + \underbrace{(\mathbf{q} \times \mathbf{x})}_{\bar{\mathbf{x}}}{}^T \mathbf{n}_i = 0 \,,$$

using the cyclic permutation identity for triple scalar products: $(\mathbf{n}\times\mathbf{x})^{\mathrm{T}}\mathbf{p} = (\mathbf{p}\mathrm{x}\mathbf{n})^{\mathrm{T}}\mathbf{x}$. Of course, *not every* linear equation system constructible from Plücker lines according to (5.44) has a nontrivial solution that indeed describes a 3D line. Lines satisfying (5.44) are said to form a *linear complex* in the terms of *line geometry* [OPW05].

Equation (5.42) is a special case of more general symmetries exhibited by points under *equiform motions*, including helical or spiral motions as important examples [OPW05][HOP05]. An equiform motion maps a point $\mathbf{x} \in \Re^3$ according to $\mathbf{y}(t) = \alpha(t)\mathbf{A}(t)\cdot\mathbf{x} + \mathbf{a}(t)$, where $\alpha(t)$, $\mathbf{A}(t)$, $\mathbf{a}(t)$ denote time-dependent scaling, rotating, and translating transformations, respectively. The mapping over a time interval describes a smooth curve in $\Re^3$, and, by introducing two-dimensional motion parameters, a surface is generated. The velocity vector of the moving point is *tangential* to this curve. As the mentioned transformations are nonsingular (invertible), the velocity vector $v(\mathbf{y})$ is a function of $\mathbf{y}$ satisfying the linear differential equation [OPW05]

$$v(\mathbf{y}) = \dot{\mathbf{y}}(t) = \underbrace{\dot{\mathbf{A}}\,\mathbf{A}^T}_{skew-symmetric}\,\mathbf{y} + \underbrace{\frac{\dot{\alpha}}{\alpha}}_{=:\,\gamma}\,\mathbf{y} \underbrace{- \dot{\mathbf{A}}\,\mathbf{A}^T\mathbf{a} - \frac{\dot{\alpha}}{\alpha}\mathbf{a} + \dot{\mathbf{a}}}_{=:\,\bar{\mathbf{c}}} = \mathbf{c}\times\mathbf{y} + \gamma\,\mathbf{y} + \bar{\mathbf{c}} \tag{5.45}$$

For brevity, the time t is omitted. Considering the *normal* $\mathbf{n}(\mathrm{y})$ to the motion curve at an arbitrary point $\mathbf{y}$, it is orthogonal to the tangent or velocity vector $v(\mathbf{y})$ and, using (5.45), therefore satisfies

$$0 = \mathbf{n}^T\left(\mathbf{c}\times\mathbf{y} + \gamma\,\mathbf{y} + \bar{\mathbf{c}}\right) = \underbrace{(\mathbf{y}\times\mathbf{n})}_{=:\,\bar{\mathbf{n}}}{}^T\,\mathbf{c} + \mathbf{n}^T\,\bar{\mathbf{c}} + \gamma\,\underbrace{\mathbf{n}^T\mathbf{y}}_{=:\,\lambda} = (\bar{\mathbf{n}}, \mathbf{n}, \lambda)^T \cdot (\mathbf{c}, \bar{\mathbf{c}}, \gamma) \tag{5.46}$$

The path normals in (5.46) play the same role as the normal rays of a sampled surface in (5.44) and satisfy the same kind of linear relation when a seventh scalar component λ, the dot product $\mathbf{y}^T\mathbf{n}$, is added to their Plücker representation:

$$(\mathbf{y}, \mathbf{n}) \;\rightarrow\; \left(\mathbf{n} \quad \mathbf{y}\times\mathbf{n} \quad \mathbf{y}^T\mathbf{n}\right) \;=:\; (\mathbf{n} \quad \bar{\mathbf{n}} \quad \lambda) \in \Re^7 \tag{5.47}$$

Path normals under equiform motion therefore form a *linear complex* $(\mathbf{c}, \bar{\mathbf{c}}, \gamma)$.

In case of noisy data, the equalities (5.44) or (5.46) are replaced by a quadratic error function to be minimized by a line complex L [HOP05]:

$$L = \underset{(\mathbf{c},\bar{\mathbf{c}},\gamma)}{\arg\min} \;\; (\mathbf{c}, \bar{\mathbf{c}}, \gamma)^T \underbrace{\sum_{i=1}^{N} (\bar{\mathbf{n}}_i, \mathbf{n}_i, \lambda_i)(\bar{\mathbf{n}}_i, \mathbf{n}_i, \lambda_i)^T}_{\mathbf{M}} (\mathbf{c}, \bar{\mathbf{c}}, \gamma) \tag{5.48}$$

The minimizing line complex L is found as a unit eigenvector to the smallest eigenvalue of the 6x6 respectively the 7x7 covariance matrix $\mathbf{M}$. It is the same closed-form solution procedure as finding the rotation to minimize the error between corresponding feature pairs (ICP), or finding quadric coefficients to minimize squared algebraic distances from sample points.

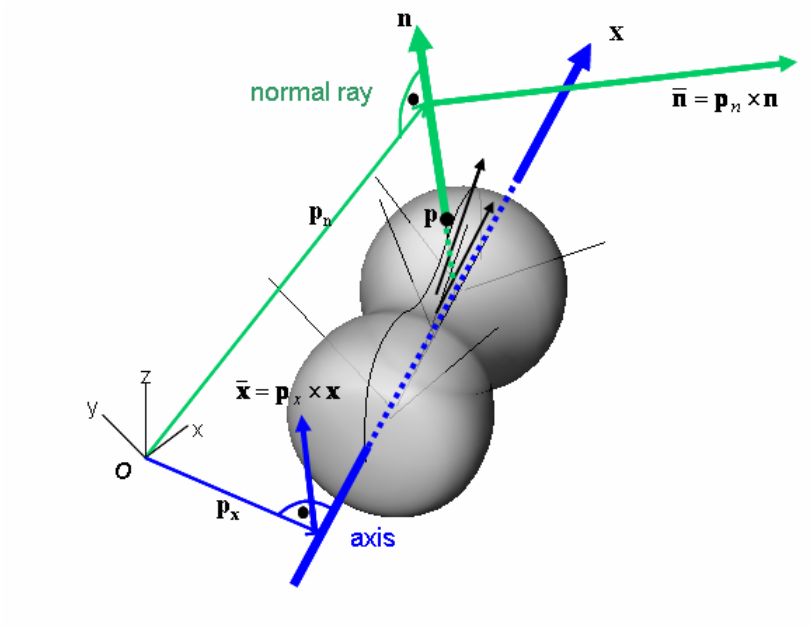

fig. 5-19 Normal rays of a surface of revolution in Plücker coordinates

Before applying (5.48) to oriented points $(\mathbf{p}_i, \mathbf{n}_i)$, the point coordinates must be *scaled* to size ≤ 1 to match the magnitude of the normal vectors as pointed out by [HOP05]. Otherwise a biased estimate would result from the covariance matrix. Of course, any point information recovered from the Plücker solution must be scaled back before use. Both forms - the 6D coordinates in (5.43) producing 6x6 covariance matrices, and the 7D coordinates (5.47) producing 7x7 matrices - are able to estimate the axis of a rotationally symmetric object from sampled points. In comparison, we can state:

- The 6D solution is sufficient for all objects having a single *straight line* of rotational symmetry and approximately circular cross sections, i.e. SoR's with arbitrary radius functions. Especially, this object class contains cones and cylinders including spherical or elliptical closures of them (e.g. tanks, vessels). The 7D solution is more general and discerns in addition not only planes and spheres but various spiral and helical surface types which, however, rarely occur in industrial installations. On the other hand, generalized cylinders with a compound or branched axis structure can neither be classified in 7D nor in 6D by a single problem instance.

- Regarding numerical stability, the 6D method showed slight advantages: for all SoR data sets, the axis direction uniquely corresponded to the *smallest* eigenvalue. In the 7D case, more often some confusion about the ranking of the target eigenvector could arise since up to 4 eigenvalues (up to 3, excluding planes) may be similarly small. On sampled shapes having principal curvatures of similar magnitude, such as a spherical closure of a cone, the second-smallest eigenvalue often yielded the expected axis direction.

- One advantage of the 7D method is to yield a *centre point* of rotation $\mathbf{y}$ about the axis. It is a point of *zero velocity* and therefore obtained directly from (5.45):

$$0 = \dot{\mathbf{y}}(t) = \mathbf{c} \times \mathbf{y} + \gamma \mathbf{y} + \bar{\mathbf{c}} \quad \Rightarrow \quad \mathbf{y} = -\begin{bmatrix} \gamma & -c_z & c_y \\ c_z & \gamma & -c_x \\ -c_y & c_x & \gamma \end{bmatrix}^{-1} \cdot \bar{\mathbf{c}} \quad where\ \mathbf{c} = (c_x, c_y, c_z)^T, \gamma \neq 0 \qquad (5.49)$$

  If the rotation is axially symmetric, the point $\mathbf{y}$ lies *on* this axis. However, the centre points returned for cylindrical or for conical pipes with a very small opening are *vanishing points* far away from the object. Shifting the point 'inside' the object along its axis will greatly amplify directional inaccuracies, however small. In the 6D case, the minimizing solution $(\mathbf{c}, \bar{\mathbf{c}})$ will only yield the direction $\mathbf{c}$ but no centre point unless $\mathbf{c}^T \bar{\mathbf{c}} \approx 0$. Only in this particular case may a point $\mathbf{p}$ satisfying $\mathbf{p} \times \mathbf{c} = \bar{\mathbf{c}}$ be found: $\mathbf{p} = (\mathbf{c} \times \bar{\mathbf{c}}) / \mathbf{c}^T \mathbf{c}$.

Especially the last point suggests using line geometry only to estimate the axis *direction*, and determine the axis point differently. For the axis direction, our implementation invokes *both methods* - 6D and 7D line complexes - on the same data: it analyzes the two smallest eigenvalues from both systems and decides for the direction of *strongest correspondence*. The according eigenvalue rankings may be different. For example, the direction for the smallest 6D eigenvalue may be closest to (include the largest dot product with) the direction of the second-smallest eigenvalue of the 7D system.

Finding the axis point to a known axis direction is easy. The sample points and normals are projected onto a plane orthogonal to the direction and passing through the component center. Within this plane (coordinate transformation $(\mathbf{p}_i, \mathbf{n}_i) \rightarrow (\mathbf{p}'_i, \mathbf{n}'_i)$) the centre $\mathbf{cp}$ is sought minimizing the 2D distance from all projected normal rays. After a simple calculation, we find:

$$\mathbf{cp} = \arg\min_{\mathbf{x}} E(\mathbf{x})$$

$$E(\mathbf{x}) = \sum_{i=1}^{N} d_{\perp}^2(\mathbf{p}'_i, \mathbf{n}'_i, \mathbf{x}) = \sum_{i=1}^{N} \left[(\mathbf{x} - \mathbf{p}'_i)^T (\mathbf{x} - \mathbf{p}'_i) - \left((\mathbf{x} - \mathbf{p}'_i)^T \mathbf{n}'_i \right)^2 \Big/ \left(\mathbf{n}'^T_i \mathbf{n}'_i \right) \right] \qquad (5.50)$$

$$\nabla E(\mathbf{x}) = \sum_{i=1}^{N} \left(\mathbf{I}_2 - \frac{\mathbf{n}'_i \mathbf{n}'^T_i}{\mathbf{n}'^T_i \mathbf{n}'_i} \right) (\mathbf{x} - \mathbf{p}'_i) = 0 \quad \Rightarrow \quad \mathbf{x} = \left(\mathbf{I}_2 - \sum_{i=1}^{N} \frac{\mathbf{n}'_i \mathbf{n}'^T_i}{\mathbf{n}'^T_i \mathbf{n}'_i} \right)^{-1} \cdot \sum_{i=1}^{N} \left(\mathbf{p}'_i - \frac{\mathbf{p}'^T_i \mathbf{n}'_i}{\mathbf{n}'^T_i \mathbf{n}'_i} \mathbf{n}'_i \right)$$

The 2D solution point $\mathbf{x}$ (5.50) projected back into 3D space gives the desired axis midpoint. If the initial direction obtained is accurate up to a few degrees, the derived point estimates will be satisfactory as well; three examples of moderate difficulty (flue gas washer, flash tank) are shown in figure 5-20. In the following sections and in chapter 6, we denote as 'the Plücker axis solution' the application of PCA to the normal ray complex (5.48), yielding the axis direction, followed by the projection (5.50) step to find the axis point.

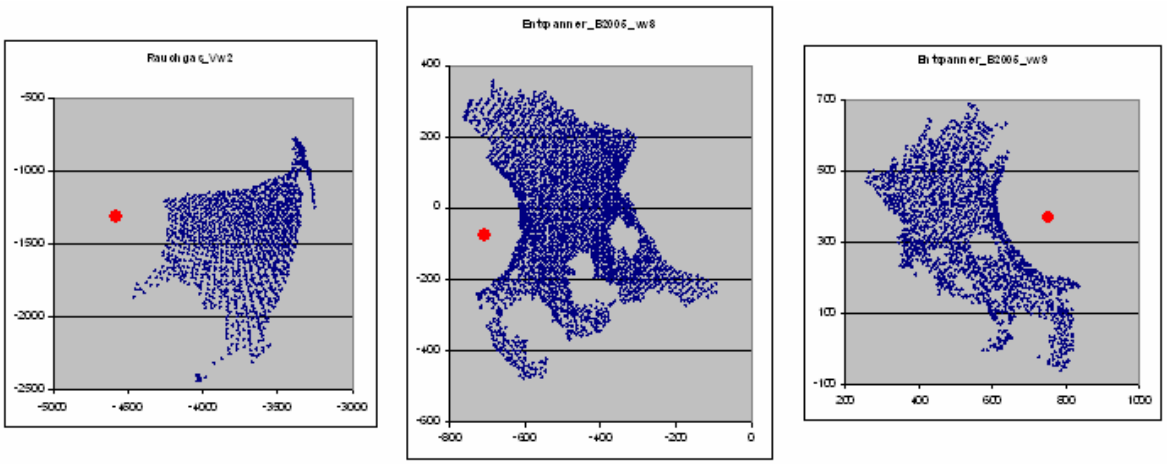

fig. 5-20 Examples of axis midpoint estimation by 2D ray projection; coordinates refer to the axis-orthogonal plane; the red bullet marks the midpoint position and the blue dots denote region points.

5.4.5 Symmetry guided partitioning

Forming straight-axis hypotheses alone, in general, is not sufficient for finding the axial symmetries in industrial installations. Components are often under-segmented; a tubular surface with a complex curved or branched rotation axis exhibits no rotational symmetry as a whole, only its parts do individually. In other cases, the part shape may be simple enough but the principal symmetry detected does not return the axis intended. For a toroidal bend of pipe, for example, an approximation of the torus axis may result where some curved medial axis *through* the pipe would be desired (cf. section 3.1 on representation). It is necessary to test and evaluate the degree of rotational symmetry on compound shapes, using different and additional criteria to the error function (5.48). Some decision criterion is needed to accept or reject the 'straight-axis'-hypothesis and to partition the point set in a reasonable way. Then the 'Plücker' method can be recursively invoked on the subsets.

This top-down strategy is consistent with our data-redundant approach on components and is similar to the principal curve splitting algorithm in section 5.3.4. Alternatively, a bottom-up strategy may be appropriate grouping smaller into larger line complexes based on some similarity measure of the eigenvalue systems as proposed by Gelfand and Guibas [GeG04][8]. Its potential towards real-time performance appears attractive but has not yet been explored on our data sets. For plant mapping application, eigenvalue similarity would need to be augmented by terms for the similarity of the axis parameters and the radii themselves.

Two combined measures of rotational symmetry were found adequate for plant mapping:

[8] The paper [GeG04] studies similar types of equiform motion as Pottmann's work [HOP05] and leads to the same error function as (5.49, in 6D). But the derivation is different: the eigen-modes are interpreted in [GeG04] as degrees of motion freedom under which the scanned shape becomes *slippable*. [HOP05] suggest a segmentation method as well: they perform RANSAC rounds on random minimal data sets of size 7 to estimate the kinematic parameters, and calculate the supporting points. In the terminology of section 3.2, this is rather a model-redundant procedure. With real-time SLAM as a primary goal, the bottom-up segmentation in [GeG04] appears to be a more promising starting point.

1. Radial-symmetric circular cross sections: each point set falling into the same projection bin h should form a uni-modal radius distribution sharply peaked at the constant radius $r[h]$ of that cross section with a standard deviation predicted from the scanning uncertainty and the bin population size. This condition will be violated if, for example, a toroidal pipe bend is parameterized about the *torus* axis: the radius distribution, for most projection intervals, will have two separate peaks (figure 5-21 on the left). The radial symmetry is also violated if the profile curve (generatrix) is closed but the rotation axis lies *outside*.

2. Uni-modal NPA angle distribution: the angles between a point **n**ormal and the **p**oint-to-**a**xis connection vector (called NPA angles) are considered in the *cross section* plane perpendicular to the axis. Each normal ray is projected into the plane (figure 5-21 on the right). For all cross sections circular and well-centred with respect to the axis, the NPA angle distribution must be uni-modal and sharply peaked in the angle interval $[0,\pi]$, near 0 $[rad]$ for convex and near π for concave SoR. Figures 5-22 to 5-24 illustrate NPA angle histograms for different component regions and axis parameters. A toroidal bend of pipe parameterized about the *torus* axis typically develops *two* angle peaks (fig. 5-22f), or, due to noise, the angle distribution will be spread rather uniformly over the entire interval $[0, \pi]$ because normal vectors almost perpendicular to the cross section plane include random angles with the point-to-axis connection. The degree of peakedness is specified by low (10%) and high percentiles (80%, 90%) of the angle distributions; the corresponding angle values for 'good' straight-axis hypotheses were obtained from training images. Employing kurtosis would be another possible measure of histogram peakedness.

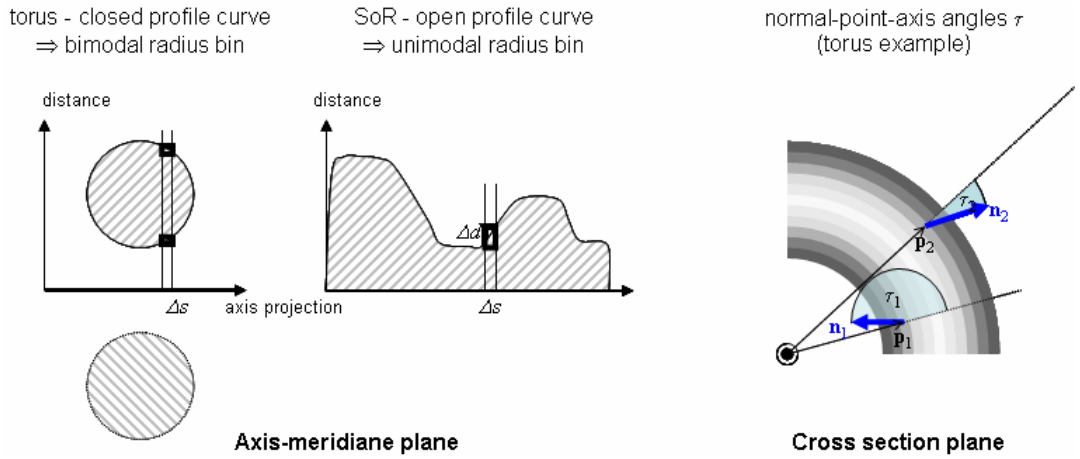

fig. 5-21 Degree of rotational symmetry considering the shape of the radius distribution (left) and the distribution of the normal-point-axis angles (right).

Criteria 1 and 2 are currently used only to accept or reject the straight-axis hypothesis, not to decide where and how to split the point set. Assuming that the straight-axis Plücker hypothesis *failed*, the next most reliable axis estimate to fall back on is actually the *tangent direction* of the principal frame; it is available from the point set at no extra cost (section 5.4.1). Using the tangent line passing through the axis point (5.51) as a substitute axis, we build the binned radius (point-to-axis distance) distribution as explained in section 5.2.4, and select the bin of maximum distance for splitting the point set. This simple strategy proved effective and efficient in locating the prominent inflexion and branch points of a compound axis.

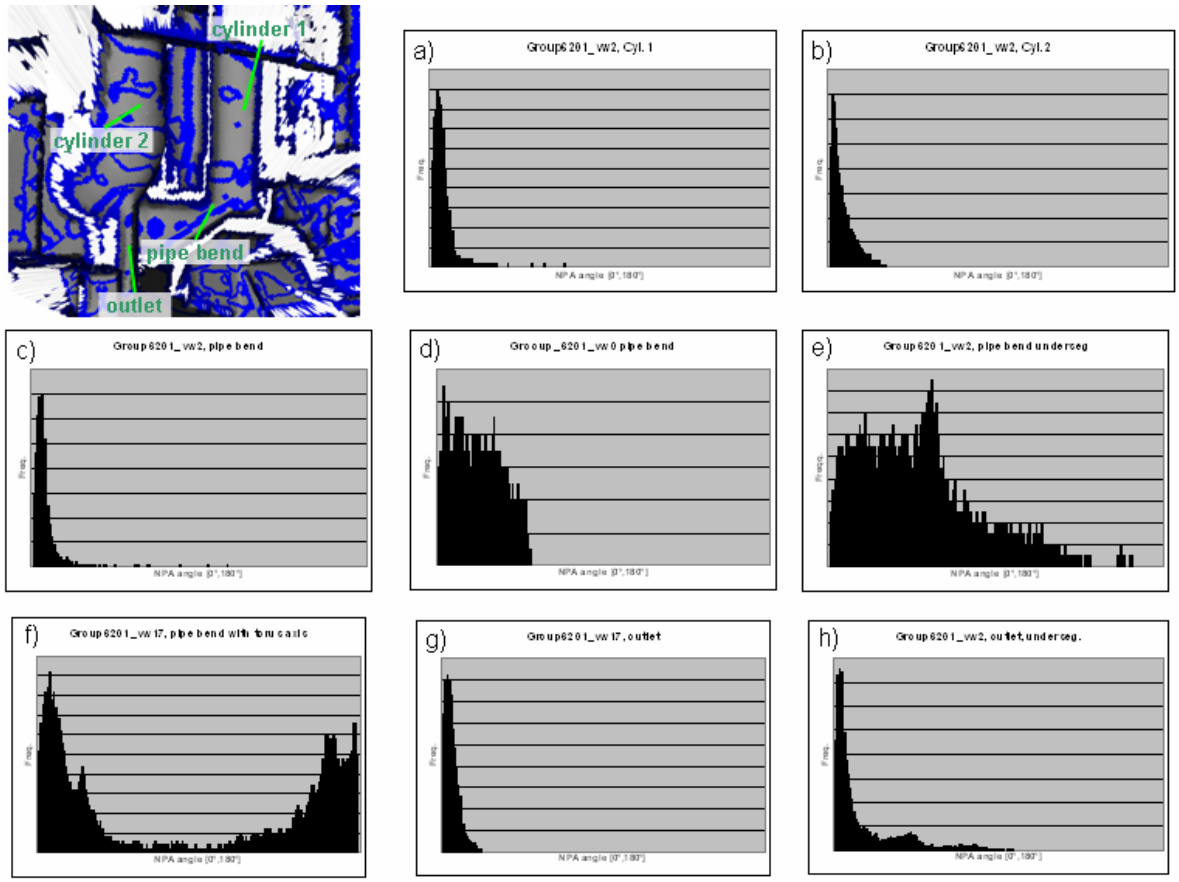

fig. 5-22 NPA angle histograms for various cylindrical regions in range view Group6201_vw1 shown on the upper left side. a) Cylinder 1, b) Cylinder 2, c) Straight section of pipe bend; d), e) Under-segmented bent pipe section, f) pipe bend parameterized with respect to a torus axis (Group6201_vw17), g) Outlet pipe properly segmented, h) Outlet pipe slightly under-segmented.

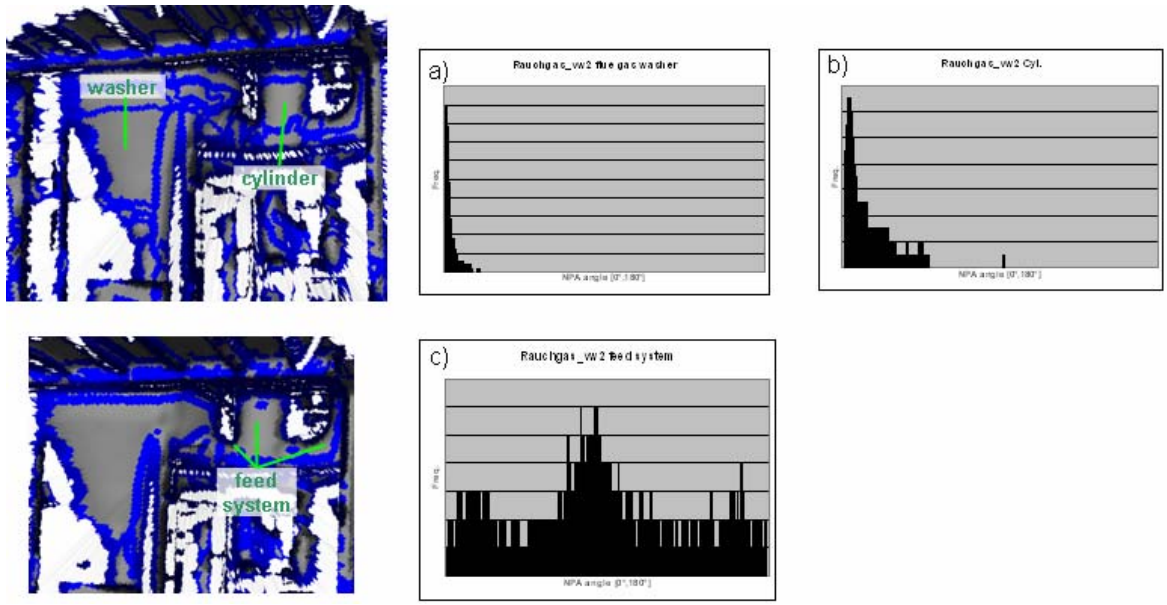

fig. 5-23 More NPA angle histograms for object regions in range view Rauchgas_vw2 shown on the left: a) Conical washer b) Vertical cylinder c) Underseg-mented feed system.

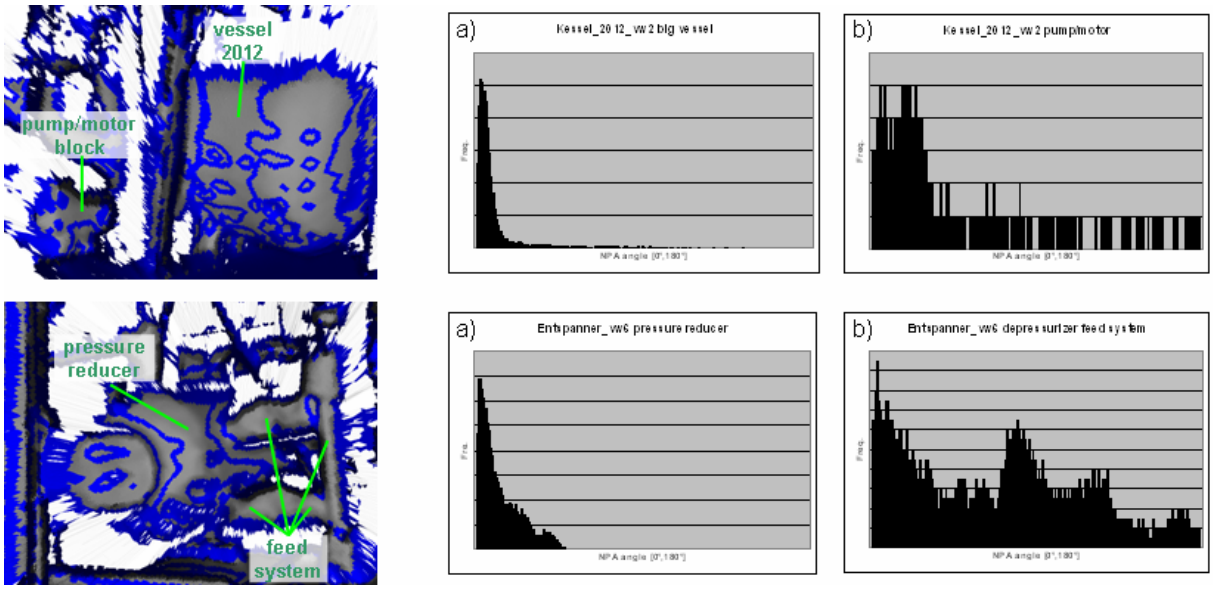

fig. 5-24 More NPA angle histograms from range views Kessel_2012_vw2 (top row: a) Vessel b) Pump-and-motor block), and Entspanner_2005_vw6 (bottom row: a) Pressure reducer unit b) Undersegmented feed system).

5.5 Parameter optimization by EM

So far, several algorithms have been proposed for generating model hypotheses from range images. These initial model representations take the form $\mathbf{K}^{(0)}$:= $\{K_1, \ldots, K_n, K_*\}$ where each K_j denotes a parameter vector modeling a SoR hypothesis or one of its special cases. To each hypothesis K_j also belongs an initial set of points $P_j^{(0)}$ supporting it. One additional 'dummy' model K_*, the non-SoR hypothesis is introduced since most range views from the work space will have points captured on surfaces not being rotationally symmetric, such as planar ones, support structures like beams and slabs, and possibly moving objects or people. The goal is to *simultaneously optimize* the parameters K_j and the assignment of point sets P_j such that measured data and model parameters best match each other. A variable number n of hypotheses is generated but for the optimization stage n is fixed, i.e. no new hypotheses will be created but existing ones may disappear.

An expectation-maximization (EM) *like* algorithm is employed for optimization. At first, the likelihoods c_{ij} are determined for each point $\mathbf{p}_i$ corresponding to the j-th model K_j. This *expectation step* requires a Bayesian sensor model predicting the measurement probabilities for known object models. The second *maximization step*, knowing now the assignment probabilities c_{ij}, finds the parameter values K_j maximizing the measurement probabilities. This is equivalent to minimizing a fitting error. Both steps alternate[9] until *convergence* to a *local optimum* in the huge optimization space which is a product of $n+1$ parameter spaces and the set of image partitions into point sets. The algorithm has been inspired by Liu's early work on 3D SLAM [Liu01] who used an EM for indoor mapping by means of infinite planes. We follow their general approach but a few major differences apply to our case of SoR mapping.

[9] The upper iteration or version index $^{(k)}$ will be dropped in the following to keep notation simple.

- First, the generated models seem to be most useful if the objects have a *finite extent* and a bounding box. The drawback that, due to partial occlusion, a contiguous object may thereby be split into fragments is remedied at a higher level: by relating *model primitives* and assigning them to *groups* of similar objects, see section 5.7. In the expectation step, bounded SoR's are handled by minor technical adaptations of the measurement probabilities.

- Second, unlike [Liu01] exploiting the linear plane constraint $(\mathbf{p}_i\text{-}\mathbf{p})^T\mathbf{n} = 0$, SoR fitting and even cone fitting admit no known linear solution [MLM01][Pet02][Tay04][Rab06]. A nonlinear least-squares optimization algorithm, being itself iterative, must be plugged into the maximization part; it works here in an *interleaved* way and concurrently with the EM.

- Third, having real-time capability in mind, not in every iteration will *each* model K_j compete for *all* points $\mathbf{p}_i$. The assignment problem is decoupled into *disjoint* or *overlapping domains* of support for each primitive. Topological connectivity is also required. Formulating spatial dependencies between measurements in a Bayesian framework ('*Data point x is more likely assigned to K if x is 4-connected to some y assigned to K*') yet avoiding circular definitions requires Markov Random Fields or Associative Markov Networks [Ang05] or a similar approach not considered here. Replacing 'assigned' by 'previously assigned' in an *iterative sequential* optimization framework however avoids definition problems. But it comes at the price of some arbitrariness of results depending on the execution order, and some greediness of assignment.

5.5.1 Expectation Step (*E-Step*)

From a probabilistic viewpoint, the noisy point measurements $\mathbf{p}_i$ are random variables, and so are all derived or dependent model parameters K_j and also the Boolean assignment variables c_{ij}. Therefore, distributions and expected values of these random variables exist. Formally, the expected value $E(c_{ij}) \in [0,1]$ equals the *posterior* probability $p\,(c_{ij}=1 |\, \mathbf{p}_i, \mathbf{K})$ that the i-th point belongs to the j-th primitive, knowing all measurements $\mathbf{p}_i$ and model parameters $\mathbf{K}$. Actually, only the inverse measurement probability $p(\mathbf{p}_i |\, c_{ij},\, K_j)$ that a point from a known primitive K_j produces a measurement $\mathbf{p}_i$ can be predicted using a sensor model. As usual, this prior probability will then be 'inverted' using Bayes' rule to give the desired posterior. Measurement probability is a conjunction of four events assumed statistically independent and normally distributed for simplicity, hence a product of four simpler probabilities. The sensor model for a point sample will be detailed now. Learning its internal parameters, adapting to different sensor properties and object environments, is however not treated at this point; some experience with the configuration parameters is reported in chapter 6.

Curvature compatibility B_{ij} assesses the likelihood of the local curvature type k_i measured at $\mathbf{p}_i$ given the curvature type of the model according to the parameter vector K_j. Different variants are conceivable. Being most restrictive, we may set $B_{ij}=1$ if k_i has type RIDGE and K_j is convex, or k_i has type VALLEY and K_j is concave, and set $B_{ij}=0$ otherwise. Accounting for uncertainty of local curvature classification, adjacent types to RIDGE such as PEAK and SADDLE_RIDGE, or PIT and SADDLE_VALLEY adjacent to VALLEY, only degrade the compatibility to $B_{ij}=0.5$. Adjacencies in the Koenderink shape space are handled similarly as adjacencies of curvature sign combinations.

Distance probability: The probability of the orthogonal point distance $d_\perp(\mathbf{p}_i, K_j)$ from the SoR surface is modeled as a zero-mean Gaussian:

$$p_d\left(\mathbf{p}_i \middle| K_j\right) \propto \frac{1}{\sqrt{2\pi}\sigma_d} e^{-\frac{d_\perp^2}{2\sigma_d^2}} \tag{5.51}$$

Projection probability: As the model primitives are *bounded* SoR segments, a point measurement $\mathbf{p}_i$ being assigned to K_j becomes less likely if its projection s_i onto K_j's rotation axis falls outside the current *interval* of projection $[s_{min}^{(j)}, s_{max}^{(j)}]$. Again, the probability of point measurements follows a zero-mean Gaussian distribution of *projection distances*. A zero projection *distance* d_s is assumed inside the projection interval ($s_{min}^{(j)} \le s_i \le s_{max}^{(j)}$) and a linearly increasing one outside, at a slope inversely proportional to the SoR radius at the respective interval end ($r_{min}^{(j)}$ or $r_{max}^{(j)}$). Therefore,

$$p_\lambda\left(\mathbf{p}_i \middle| K_j\right) \propto \frac{1}{\sqrt{2\pi}\sigma_s} e^{-\frac{d_s^2}{2\sigma_s^2}} \quad where \qquad d_s\left(\mathbf{p}_i, K_j\right) = \max\left(0, \frac{s_{min}^{(j)} - s_i}{r_{min}^{(j)}}, \frac{s_i - s_{max}^{(j)}}{r_{max}^{(j)}}\right) \tag{5.52}$$

Angular probability depends on two distance terms: the *normal angle* deviation and the *radial angle* deviation. The former one, denoted $d_{\alpha N}$, measures the deviation of the angle included between point normal $\mathbf{n}_i$ and axis $\mathbf{a}_j$ from the *desired* angle which should be $(\pi+\delta_j)/2$ for a convex SoR, and $(\pi-\delta_j)/2$ for a concave SoR; δ_j being the opening angle near the corresponding SoR point known from the radius function of K_j (3.7). The latter term, $d_{\alpha R}$, assesses the radial angle in a *plane* orthogonal to the axis: between the point-to-axis projection vector $\mathbf{p}_{i,\perp}$ and the vector $\mathbf{n}_i \times \mathbf{a}^{(j)}$ which is orthogonal to the projected normal $\mathbf{n}_i$. The radial angle deviation is very closely related to the normal-point-axis (NPA) angle from section 5.4.5 by an offset angle of $\pi/2$. A zero NPA angle corresponds to a radial angle of $\pi/2$ and an angle cosine of 0, i.e. corresponds to a zero distance. In accordance with the actual implementation, angular distances are expressed in terms of angle cosines, i.e. dot product of unit vectors:

$$\begin{aligned} p_\alpha\left(\mathbf{p}_i \middle| K_j\right) &\propto \frac{1}{\sqrt{2\pi}\sigma_\alpha} e^{-\frac{d_{\alpha N}^2 + d_{\alpha R}^2}{2\sigma_\alpha^2}} \\ d_{\alpha N}\left(\mathbf{p}_i, K_j\right) &= \begin{cases} \mathbf{n}_i^T \mathbf{a}_j + \sin\left(\delta_j/2\right) & if\ K_j\ convex \\ \mathbf{n}_i^T \mathbf{a}_j - \sin\left(\delta_j/2\right) & if\ K_j\ concave \end{cases} \\ d_{\alpha R}\left(\mathbf{p}_i, K_j\right) &= \overline{\mathbf{d}}_{i,j,\perp}^T \cdot \overline{\mathbf{p}}_{i\perp} \quad where \quad \overline{\mathbf{d}}_{i,j,\perp} := \frac{\mathbf{n}_i \times \mathbf{a}_j}{\left\|\mathbf{n}_i \times \mathbf{a}_j\right\|}, \quad \overline{\mathbf{p}}_{i\perp} := \frac{\mathbf{p}_{i,\perp}}{\max\left(\left\|\mathbf{p}_{i,\perp}\right\|, \varepsilon_d\right)} \end{aligned} \tag{5.53}$$

The lower bound ε_d in the denominator of the vector $\mathbf{p}_{i,\perp}$ in (5.53) counteracts the *sensitivity* of angular deviation as a function of the SoR radius r (which is $\sim 1/r$); without it, small point errors on narrow tubular objects would produce large radial angle deviations and thus make any measurement highly unlikely. ε_d should be chosen in the order of magnitude of the distance uncertainty, given by the standard deviation of the laser scanner measurements.

The total measurement probability for a genuine primitive K_j is the product of those four factors. Assuming *no* SoR (K_*), the measurement probability $p\ (\mathbf{p}_i | K_*)$ should be accounted for,

too. Naturally, little is known about it. [Liu01] therefore set it equal to the *unconditional* probability of a random measurement, assuming uniform density inside the laser scanner's range of measurement $[0, D_{Max}]$, and zero density outside. Combining the cases, we get

$$p(\mathbf{p}_i|K_j) \propto B_{ij}(\mathbf{p}_i|K_j)\cdot p_d(\mathbf{p}_i|K_j)\cdot p_\lambda(\mathbf{p}_i|K_j)\cdot p_\alpha(\mathbf{p}_i|K_j)$$
$$p(\mathbf{p}_i|K_*) \propto \begin{cases} \frac{1}{D_{Max}} & if\ 0 < \|\mathbf{p}_i\| < D_{Max} \\ 0 & else \end{cases} . \tag{5.54}$$

Using Bayes' rule, the desired posterior probability is obtained from the measurement probabilities. Since the priors $p(\mathbf{p}_i \mid \mathbf{K})$ for a random measurement and $p(c_{ij} \mid \mathbf{K})$ for a random assignment may be assumed uniformly and identically distributed for all i and j, they are summarized by a constant factor η which is calculated from the normalization constraint.

$$E(c_{ij}) = p(c_{ij} \mid \mathbf{p}_i, \mathbf{K}) = \frac{p(\mathbf{p}_i \mid c_{ij}, \mathbf{K})\cdot p(c_{ij} \mid \mathbf{K})}{p(\mathbf{p}_i \mid \mathbf{K})} = p(\mathbf{p}_i \mid c_{ij}, \mathbf{K})\cdot\eta = p(\mathbf{p}_i \mid K_j)\cdot\eta \tag{5.55}$$

$$\sum_{j=1,\cdots,n,*} c_{ij} = 1 \quad\Rightarrow\quad \sum_{j=1,\cdots,n,*} E(c_{ij}) = 1 \quad\Rightarrow\quad \eta = \frac{1}{\sum_{j=1,\cdots,n,*} p(\mathbf{p}_i \mid K_j)} . \tag{5.56}$$

5.5.2 Maximization Step (*M-Step*)

Knowing the expected assignments c_{ij}, the M-step goal is to find the model parameter values K_j maximizing the total measurement probabilities. For real-time feasibility, several simplifications are made. Firstly, *only one* probability term (5.5.1), the geometric distance, is considered in the M-step. As usual, the logarithm of probabilities is minimized, leading to a weighted geometric fitting error or *cost function* of the unknown SoR parameters which is minimized:

$$\mathbf{K} = \underset{(K_1,\ldots,K_n)}{\arg\min} \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{n} E(c_{ij})\cdot d_\perp^2(\mathbf{p}_i, K_j)}_{Cost(K_1,\ldots,K_n)} \quad where \qquad E(c_{ij}) = p(c_{ij} = 1 \mid \mathbf{p}_i, K_j) \tag{5.57}$$

Secondly, use is made of the fact that for any point the assignment probabilities during EM quickly become dominated by a single K_j or by the non-SoR object K_*. Therefore, the hypotheses generate a partition $\{P_j\}$ into supporting point sets. The joint costs (5.57) are decomposed into $n+1$ cost terms and minimized independently:

$$P_j := \left\{ i : E(c_{ij}) >> \sum_{k\neq j} E(c_{ik}) \right\} \quad (j = 1,\ldots,n,*) \quad\Rightarrow\quad Cost(K_j) \approx \sum_{i\in P_j} E(c_{ij})\cdot d_\perp^2(\mathbf{p}_i, K_j)$$
$$K_{j,\min} = \underset{K_j}{\arg\min}\ Cost(K_j) \quad (j = 1,\ldots,n) \tag{5.58}$$

Each parameter vector K_j minimizes the (sum of squared) geometric distances within its own support set. The assignment probabilities $E(c_{ij})$ known from the E-step are kept constant and

serve as weighting factors, or may even be ignored if already close to 1. The mean of $E(c_{ij})$ within a support set P_j will be referred to as the *hypothesis quality* q_j and the sum as the *hypothesis weight* w_j. The minimization (5.58) applies to two types of primitives:

- a *truncated cone* segment described by six parameters and, in addition, the sign (convex or concave) and the interval of projection values, or
- a *surface of revolution* (SoR) characterized by a straight axis and a piecewise linear radius function.

The SoR is regarded as a sequence of truncated cones coerced to share the same axis. In principle, any nonlinear numerical optimization algorithm for cones may be used to optimize a subset of free parameters and keep the remaining ones fixed. However, estimating a piecewise linear radius function would require additional constraints ensuring continuity at the break points. In addition, the point set would need to be partitioned according to different interval sections, which we avoid. An alternative is to separate the parameters into *pose* (axis) and *shape* (radius function) parameters, and apply an ICP-like algorithm [BeM92] to estimate the pose by a *rigid transformation*. This can be done using *any* radius function if only a suitable closest-distance function to the surface is provided. Conversely, knowing the axis, an approximation of the radius function is obtained as described before in section 5.2.4. Both steps are alternated or interleaved inside the EM loop without affecting the numerical performance (accuracy, stability) much. In summary, the following sub-sections will present

- A nonlinear LM (Levenberg-Marquardt) algorithm to simultaneously optimize *all* parameters of *cones*, including the issues of parameterization and transformation,
- A (nonlinear) ICP algorithm to optimize the *axis parameters* of *SoR* with *piecewise linear* and continuous *radius functions*.

5.5.2.1 Minimal and redundant parameterization

This and the following sub-section apply to truncated cones. The first decision affects the parameter representation. A *redundant* parameterization has the advantage of being visually intuitive and easy to interpret, for example when assessing the magnitude of value changes or expressing measurement probabilities in the E-step. It is preferred for *hypothesis generation* as some constraints become *linear* and solvable in closed form that are nonlinear, otherwise (section 5.4.5). The redundant cone parameter vector has 10 scalar values

$$K_R = (\mathbf{a}, \mathbf{s}, r_0, t, s_{min}, s_{max}).$$

$\mathbf{a}=(a_x, a_y, a_z)^T$ represents a unit axis direction, $\mathbf{s}=(s_x, s_y, s_z)^T$ an *arbitrary* starting point on the axis - *not* the cone apex - serving as the *origin of projection*, i.e. $s(\mathbf{p}) = (\mathbf{p}\text{-}\mathbf{s})^T\mathbf{a} \Rightarrow s(\mathbf{s}) = 0$. r_0, t denote *intercept* respectively *slope* of the radius as a function of projection s, i.e. $r(\mathbf{p}) = r(s(\mathbf{p})) = r_0+t\cdot s(\mathbf{p})$. $[s_{min}, s_{max}]$ denotes the projection interval of the points currently assigned; $r(s_{min})$, $r(s_{max})$ are the height, respectively, base radii of a truncated cone. The cone opening angle $\delta = tan^{-1}(t)$; the normal vectors include an angle of $\pi/2+\arctan(t)$ with the axis in case of a convex cone, and $\pi/2-\arctan(t)$ for a concave one.

On the other hand, for numerical optimization (M-Step), a *minimal* set of *independent* parameters is preferred because it allows *unconstrained* minimization meaning simplicity and numerical stability. Otherwise, constraints like vectors being unit or orthogonal must be handled, e.g. using Lagrange multipliers, singularities must be addressed, and there is a higher risk of numerical instability. In this work, the minimal parameterization by Lukács and Marshall [LMM97] is adopted, which has essentially been used also by Taylor [Tay04] and others. It requires six scalar parameters:

$$K_M = (\rho,\ \varphi,\ \theta,\ \alpha,\ R,\ \delta).$$

The first three *spherical coordinates* (distance ρ, azimuth angle φ, and elevation angle θ) represent the point $\mathbf{p}_\perp$ on the axis line closest to the origin. Within the plane orthogonal to $\mathbf{p}_\perp$, the axis direction is uniquely determined by a rotation angle α, which forms the forth parameter. The fifth parameter $R \geq 0$ denotes the cone radius at the axis point $\mathbf{p}_\perp$, the sixth one the (half) cone opening angle δ. Figure 5-25 illustrates the notations and the relationship between both parameter representations.

Two transformations $\mathrm{T}_{\mathrm{R}\to\mathrm{M}}$ and $\mathrm{T}_{\mathrm{M}\to\mathrm{R}}$ are introduced to convert the two representations when alternating between E-step and M-step.

$$\mathrm{T}_{\mathrm{M}\to\mathrm{R}}:\quad \begin{aligned} \mathbf{s} &= \rho\cdot\begin{pmatrix}\cos\varphi\sin\theta & \sin\varphi\sin\theta & \cos\theta\end{pmatrix}^T \\ \mathbf{a} &= \cos\alpha\cdot\begin{pmatrix}\cos\varphi\cos\theta & \sin\varphi\cos\theta & -\sin\theta\end{pmatrix}^T + \sin\alpha\cdot\begin{pmatrix}-\sin\varphi & \cos\varphi & 0\end{pmatrix}^T \\ r_0 &= R,\quad t = \tan\delta \end{aligned} \tag{5.59}$$

$$\mathrm{T}_{\mathrm{R}\to\mathrm{M}}:\quad \begin{aligned} \mathbf{p}_\perp &= \mathbf{s} - \left(\mathbf{s}^T\mathbf{a}\right)\cdot\mathbf{a} =: \begin{pmatrix}p_x & p_y & p_z\end{pmatrix}^T \Rightarrow \quad \rho = \sqrt{p_x^2+p_y^2+p_z^2}, \\ & \varphi = \tan^{-1}\left(\frac{p_y}{p_x}\right) \in [-\pi/2,\pi/2],\quad \theta = \begin{cases}\cos^{-1}(p_z) & \text{if } \operatorname{sgn}(a_z)\geq 0 \\ -\cos^{-1}(p_z) & \text{else}\end{cases} \in [-\pi,\pi] \\ & \alpha = \tan^{-1}\left(\cos\theta\cdot\frac{\cos\varphi\, a_y - \sin\varphi\, a_x}{\cos\varphi\, a_x + \sin\varphi\, a_y}\right) \in [-\pi/2,\pi/2] \\ & R = r_0 - \left(\mathbf{s}^T\mathbf{a}\right)\cdot t,\quad \delta = \tan^{-1}(t). \end{aligned} \tag{5.60}$$

Converting semantically different representations (K_M "$\subset$" K_R) one of which is unique and the other one is not, involves a few subtleties from the software implementation point of view. (Similar problems occur in tracking 3D orientation representations, such as quaternions and Euler angles, noted e.g. by Welch [Wel97]). Before executing $\mathrm{T}_{\mathrm{R}\to\mathrm{M}}$, check that a subsequent inverse transformation $\mathrm{T}_{\mathrm{M}\to\mathrm{R}}\bullet\ \mathrm{T}_{\mathrm{R}\to\mathrm{M}}$ will restore the *redundant* part of K_R correctly, and, if needed, flip the axis direction $\mathbf{a}$. When applying $\mathrm{T}_{\mathrm{M}\to\mathrm{R}}$ after *changes* to K_M have been made, update the *extra* information in K_R (not represented in K_M) consistently with its state *before* K_M was changed[10].

[10] A typical simple function needed *normalizes* the projection interval not represented in K_M and the according radii values: $(r(s),\ s_{min},\ s_{max})) \to (r(s+s_{min}),\ 0,\ s_{max}\text{-}s_{min}))$

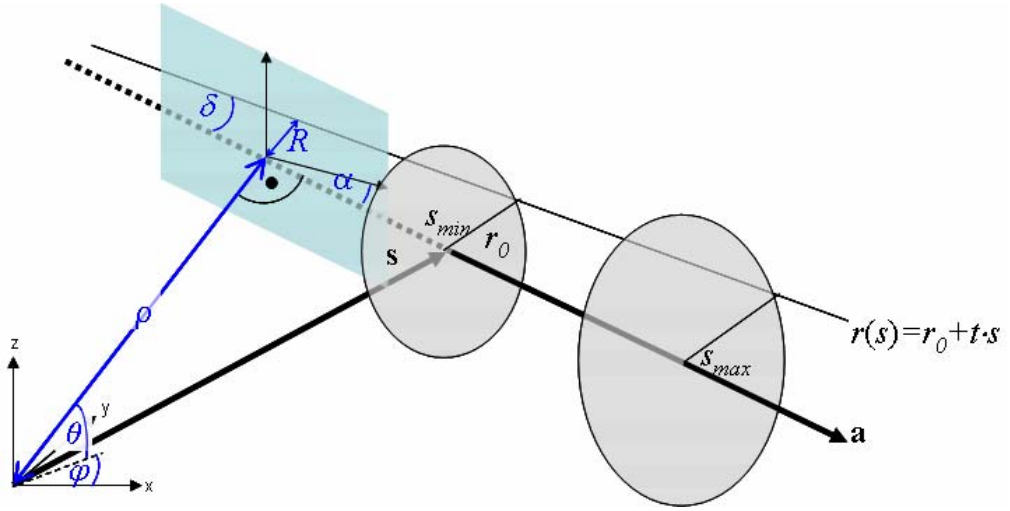

fig. 5-25 Relationship between minimal and redundant parameter representation

5.5.2.2 LM optimization

The Levenberg-Marquardt [PrF88] implementation for cone parameter fitting is summarized. First, the gradient vector of the error with respect to the six input parameters needs to be calculated. Notations **s**, **a** are used to denote position **s**(ρ, φ, θ) resp. direction **a**(φ, θ, α) vectors calculated (eq. 5.59), and $\Delta\mathbf{p}_i := \mathbf{p}_i - \mathbf{s}$. The signed geometric distance $d_\perp(\mathbf{p}_i, K_M)$ of a point $\mathbf{p}_i$ from the cone surface described by K_M in (5.58) according to eq. (3.8) is:

$$d_\perp(\mathbf{p}_i, K_M) = \left(d(\mathbf{p}_i, L(\mathbf{s}, \mathbf{a})) - R\right)\cdot\cos\delta - \left(\Delta\mathbf{p}_i^T\mathbf{a}\right)\cdot\sin\delta \quad . \tag{5.61}$$

Its gradient vector $\nabla d_\perp = \partial\, d_\perp(\cdot,\cdot) / \partial K_M$ with respect to parameters $K_M = (\rho,\varphi,\theta,\alpha,R,\delta)^{\mathrm{T}}$ evaluated at point $\mathbf{p}_i$ reads, using product and chain rules:

$$\nabla d_\perp(\mathbf{p}_i) = -\frac{\cos\delta}{d_\perp(\mathbf{p}_i, K_M)}\cdot\mathbf{u}_i - \left(\frac{\left(\Delta\mathbf{p}_i^T\mathbf{a}\right)\cdot\cos\delta}{d_\perp(\mathbf{p}_i, K_M)} + \sin\delta\right)\cdot(\mathbf{w}_i - \mathbf{v}) - \mathbf{r}_i \quad \textit{where:}$$

$$\mathbf{u}_i = \Delta\mathbf{p}_i^T\nabla\mathbf{s}, \quad \mathbf{v} = \mathbf{a}^T\nabla\mathbf{s}, \quad \mathbf{w}_i = \Delta\mathbf{p}_i^T\nabla\mathbf{a}, \quad \textit{and} \tag{5.62}$$

$$\mathbf{r}_i = \left(0 \quad 0 \quad 0 \quad 0 \quad \cos\delta \quad \frac{\Delta\mathbf{p}_i^T\mathbf{a}}{\cos\delta} - \sin\delta\right)^T \qquad (1 \le i \le N)$$

The auxiliary gradients used and referred to in (5.62), $\nabla\mathbf{s}$ of the starting point and $\nabla\mathbf{a}$ of the axis direction

$$\nabla\mathbf{s} := [\partial\mathbf{s}/\partial\rho, \quad \partial\mathbf{s}/\partial\varphi, \quad \partial\mathbf{s}/\partial\theta, \quad \partial\mathbf{s}/\partial\alpha, \quad \partial\mathbf{s}/\partial R, \quad \partial\mathbf{s}/\partial\delta\,]^{\mathrm{T}} \in \Re^{3\times6} \text{ and}$$

$$\nabla\mathbf{a} := [\partial\mathbf{a}/\partial\rho, \quad \partial\mathbf{a}/\partial\varphi, \quad \partial\mathbf{a}/\partial\theta, \quad \partial\mathbf{a}/\partial\alpha, \quad \partial\mathbf{a}/\partial R, \quad \partial\mathbf{a}/\partial\delta]^{\mathrm{T}} \in \Re^{3\times6}$$

are listed component-wise in the following tabular formula (5.63) where $c\cdot$ abbreviates $\cos(\cdot)$ and $s\cdot$ abbreviates $\sin(\cdot)$:

$$
\begin{array}{lllll}
 & \partial\mathbf{s}/\partial(\cdot) & \partial\mathbf{a}/\partial(\cdot) & & \\
\partial(\cdot)/\partial\rho & (c\varphi\, s\theta,\, s\varphi\, s\theta,\, c\theta)^T & \mathbf{0} & & \\
\partial(\cdot)/\partial\varphi & \rho(-s\varphi\, s\theta,\, c\varphi\, s\theta,\, 0)^T & (-s\varphi\, c\theta\, c\alpha - c\varphi s\alpha, & c\varphi\, c\theta\, c\alpha - s\varphi s\alpha, & 0\quad)^T \\
\partial(\cdot)/\partial\theta & \rho(c\varphi\, c\theta,\, s\varphi\, c\theta,\, -s\theta)^T & (-c\varphi\, s\theta\, c\alpha, & -s\varphi s\theta\, c\alpha, & -c\theta\, c\alpha)^T \\
\partial(\cdot)/\partial\alpha & \mathbf{0} & (-c\varphi\, c\theta\, s\alpha - s\varphi c\alpha, & -s\varphi\, c\theta\, s\alpha + c\varphi c\alpha, & s\theta\, s\alpha\quad)^T \\
\partial(\cdot)/\partial R & \mathbf{0} & \mathbf{0} & & \\
\partial(\cdot)/\partial\delta & \mathbf{0} & \mathbf{0} & &
\end{array}
\tag{5.63}
$$

For the entire cost function (5.58), the gradient vector becomes

$$\nabla Cost = \frac{2}{N}\sum_{i=1}^{N} E(c_i)\cdot d_\perp(\mathbf{p}_i, K_M)\cdot\nabla d_\perp(\mathbf{p}_i). \tag{5.64}$$

In addition to the gradient, the LM algorithm also uses the 6x6 Hessian matrix of second partial derivatives $(\mathbf{H}_{kl}) = (\partial^2 d_\perp / \partial K_M[k]\, \partial K_M[l])_{1\le k,l\le 6}$ and approximates it by products of first-order derivatives [PrF88]:

$$\widetilde{\mathbf{H}}_{kl} = \sum_{i=1}^{N} \nabla d_\perp(\mathbf{p}_i)[k]\cdot\nabla d_\perp(\mathbf{p}_i)[l] \quad (1\le k,l\le 6) \tag{5.65}$$

The Hessian approximant matrix together with an adaptive scaling factor λ determines direction and magnitude of the parameter step towards the local optimum:

$$K_M^{(m+1)} = K_M^{(m)} - \left(\widetilde{\mathbf{H}} + \lambda^{(m)}\cdot diag\left(\widetilde{\mathbf{H}}_{11},\cdots,\widetilde{\mathbf{H}}_{66}\right)\right)^{-1}\nabla Cost \tag{5.66}$$

Factor $\lambda^{(m)}$ in the LM iteration (5.66) depends on the current function value: if the cost decreases, the current step m is accepted and λ *decreased* by a factor *10*, thereby *enlarging* the step size via the matrix inversion. The step direction then resembles a Gauss-Newton step ($-\mathbf{H}^{-1}\cdot\nabla Cost$). If the cost increases, the current step is retracted and λ *increased*, thereby making the Hessian diagonally dominant (nonsingular), *shortening* at the same time the step size and proceeding more in the direction of steepest descent. A starting value $\lambda_0=10^{-3}$, minimum and maximum values $\lambda_{min}=10^{-7}$ and $\lambda_{max}=10^{3}$ were used. When λ exceeds the value λ_{max} while the error still increases the algorithm terminates unsuccessfully. This may happen if the initial parameter values are not close to a minimum or if the step size is too large (overshooting), or if the H-matrix approximation becomes singular. If consecutive iterations decrease the error by less than 0.1% it terminates successfully.

Despite step control, the magnitude of parameter changes (5.66) may become unreasonably large from a physical point of view. The first (ρ) and the fifth parameter (R) are distances and relate to the root σ_E of the mean squared distances in the cost function (5.58); the remaining angle parameters relate to the constant angle π, a fraction of which e.g. $\Delta\alpha=\pi/4$ bounds any reasonable changes. If the magnitude of a parameter step exceeds certain thresholds, we record a 'LM is banging around' situation: the algorithm probably jumped to a different local minimum - it may still converge - and the start hypothesis may be invalid. This will cause the LM to terminate unsuccessfully. The magnitude is defined as follows:

$$\|\Delta K\| := \sum_{i=1,5} \frac{|K_i|}{\sigma_E} \quad + \quad \sum_{i=2,3,4,6} \frac{|K_i|}{\Delta\alpha} \tag{5.67}$$

5.5.2.3 Iterative-closest-point (ICP) optimization

Cone, or generally, SoR parameters characterize the object pose by the axis and its shape by the radius function. Fitting the axis parameters to the data amounts to finding a *rigid transformation* (rotation and translation) and may therefore be estimated e.g. by an *iterative closest point* algorithm (ICP [BeM92]). After changing the axis parameters, the radius function will be adjusted by re-projecting the data points and fitting a new line or polygon function to the (s_i, r_i) pairs (section 5.2.4).

ICP rotates and translates a set of data points until their pair-wise squared distances to a given set of model points or model surface becomes minimal. Lacking more specific criteria of correspondence, a data point is assigned to the closest point in the set or on the surface. The two steps of correspondence and transformation, each of which decreases the distance, are iterated until convergence to a local optimum of pose. For a data point with attached normal ($\mathbf{p}_i$, $\mathbf{n}_i$) the corresponding point and normal ($\mathbf{p'}_i$, $\mathbf{n'}_i$) on the model object are calculated in closed form and in constant time using the formula (3.11) explained in chapter 3. The ICP translation maps the respective centers of data and model points. To find the rotation, one may either use only the points, i.e. the center difference vectors, or also the corresponding normal directions. In the latter case, point differences should be scaled to match the unit magnitude of normal directions.

Similar to the parameter norm (5.31) for the LM, the magnitudes of the ICP rotation and translation are to be monitored; too large values indicate invalid hypotheses and should be discarded. Contrary to LM, the eigenvalue based ICP method requires no linear approximation and produces no linearization errors. Also it is easier to adapt to different model shapes. For example, if the radius function were represented by a smooth B-spline function with control parameters rather than a polygon, for ICP one would only adapt the shortest distance function, but for the LM algorithm derivatives w.r.t an unknown number of control parameters had to be supplied. A disadvantage of using ICP for cone/SoR approximation is the separation between the axis and the radius fitting; their different convergence behaviors are not well attuned to each other.

For best performance, ICP as the M-step algorithm should run few iteration loops, only, and then detach control to the E-step for updating the point set. This will be investigated experimentally in chapter 6.

5.5.2.4 The EM loop

The entire EM algorithm is summarized in a method *ParamOptimize_EM* acting on a SoR hypothesis. The hypothesis comprises a redundant parameter vector K_j and an initial support set of points P_j referring to a range image Img. Its main loop iterates the M-Step and the E-Step until termination. Output parameters are the root of the mean distance error to the fitted or partially fitted (registered) model, and the final weight attached to the hypothesis. The M-

step calls the *LocalOptimizer* method which in turn invokes LM by default for cone hypotheses and invokes ICP under the following conditions: for general SoR hypotheses, after a previous LM step terminated unsuccessfully, or if explicitly user-selected (configuration parameter). Inside the LM method, the transformations $T_{R\to M}$ into minimal representation are called at the beginning and $T_{M\to R}$ at the end. Neither LM nor ICP change the projection parameters (s_{min}, s_{max}); they are subsequently adjusted to the changed axis (*UpdateProjRadius*). The following E-step (*AssignPointDoM*) repeats calculating the posterior assignment probabilities for the currrent support set, retaining points with dominant current hypothesis ($E(c_{ij}) > th_{DoM} > 0.5$) and then dilating the set until the new support region is maximal. Mean and sum of the assignment probabilities, *Prob* and *Weight*, are returned for the support set.

Several conditions must be fulfilled for the EM algorithm to continue. Firstly, the previous M-step must have successfully terminated and the respective method decreased its fitting error; LM and ICP may produce incomparable errors. Secondly, the previous E-step must have significantly increased either the mean probability or the weight (ε_P, $\varepsilon_W \geq 0.1\%$). In the latter case, *Prob* may have decreased but the increase in points more than compensated for that. If these conditions are met, the new performance values *Prob*, *Weight* and the model parameters are saved. Two influence factors in the probability definition in section 5.5.1 progressively decrease in each loop: the ability to grow beyond the current interval of projection (*ProjGrowthFac*) and the relative importance of the normal agreement compared to the distance error (*NormalWeightFac*). Finally, the radius function is re-estimated for the new point set (*EstimateRadiusFunction*). If the current M-step or E-step deteriorated the probability or weight, the most recently saved parameters are restored and the EM loop terminates.

Algorithm *ParamOptimize_EM* (**in out** RangeImage *Img*,
out double *Error*, **out** double *Weight*)

```
do
{
    // Maximization (M) step
    Error = LocalOptimizer (Img);
    UpdateProjRadius ();

    // Expectation (E) step
    (Prob, Weight) = AssignPointDoM (Img, NormalWeightFac, ProjGrowthFac);

    if  previous M-step successful ∧ Error<CurrentError ∧
        (Prob>CurrentProb·(1+εp) ∨ Weight>CurrentWeight·(1+εw))
    {
        CurrentError = Error; CurrentProb = Prob; CurrentWeight = Weight;
        SaveModelParameters();
        AdaptWeightFactors (NormalWeightFac, ProjGrowthFac);
        EstimateRadiusFunction (Img);
    }
    else
    {
        Improving = false;
        if Prob<CurrentProb ∨ Weight<CurrentWeight
            RestoreModelParameters();
    }
} while Improving;
```

5.6 Empirical hypothesis evaluation

Distance measures indicate the error when fitting specific model parameters to point data, but not the aptitude of the model class itself in describing the data. To answer this question one must compare different model classes in explaining the same data. A principled approach from statistical data analysis would process a list of model classes, estimate parameters for each of them, calculate scores provided by an acceptance test, and select the model with highest score. An example of this approach is variable-order surface fitting [BeJ88]). Such a list should always include basic model alternatives like plane and sphere, and should compare them to the SoR hypothesis in explaining the data before taking a final decision.

Except for the simple plane and sphere models we see severe problems applying such a 'positive' model competition approach in practice:

1) Such a list of models may hardly ever be complete, viewing the dozens of GC and SoR subclasses [ZeN96] known and the lack of suitable prior knowledge of the work space.
2) A lot of wasteful testing might be done, defeating our goals of real-time SLAM and object classification which should be ultimately *functional* than based on *shape* details.
3) Processing a model list would not solve the main problem: discarding well-fitting hypotheses that do not at all constitute real, independent objects (*phantom objects*).

An alternative approach by *empirical evidence* is therefore taken: accumulate evidence that a point set comes from a genuine rotationally symmetric object (true-positive evidence), then accumulate evidence for the contrary (**false-positive evidence**), combine the pro's and con's to an overall score, sort the hypotheses by score and discard all but the few ones accumulating the highest score.

High quality and weight from the EM algorithm do provide positive evidence. For negative evidence, we use empirical knowledge of main sources of false hypotheses. On almost flat objects ridge or valley regions may generate GC hypotheses with large radii. Regions of high curvature around crease edges ('chamfers') often generate GC hypotheses with small radii, due to non-negligible mask sizes for normal and curvature estimation. In both cases, the SoR urface is smoothly *tangentially connected* to a surface patch of a different model class, probably a planar one, or is tangential to two or three adjacent planar patches. Analysis of tangentiality therefore plays a key role in accumulating evidence. Further cues are provided by the viewing angle analysis and from the degree of coverage of a region by a model surface and, vice versa, the model explanation by region points. These cues are useful not only for deciding the right model type but also for selecting at run time the most suitable estimation algorithms (section 5.8). Last but not least, the evaluation provides additional feature properties useful for the multi-view feature association and location estimation (SLAM).

Quality q, Weight w: As mentioned earlier, the SoR hypothesis quality q is the mean posterior assignment probability within its support set, and the hypothesis weight w is the sum of assignment probabilities.

Conflict, Conflict-reduced Weight: Recalling the assignment probabilities $E(c_{ij})$ from section 5.5.1 and abbreviating by p_{ij} the degree of conflict dcf and the conflict-reduced weight w_c of a hypothesis K_j are defined as

$$dcf := \frac{\sum_{i \in P_j} \sum_{k \neq j} p_{ik}}{w} < 1, \quad w_c := \sum_{i \in P_j} \underbrace{\left(p_{ij} - \sum_{k \neq j} p_{ik} \right)}_{\geq 0!} = w \cdot (1 - dcf) \tag{5.68}$$

In (5.68), the sum of probabilities of conflicting assignments is subtracted; the result remains positive as point $\mathbf{p}_i$ is assigned to K_j only if p_{ij} takes the *absolute majority*.

Component explanation: Every SoR hypothesis is created from a randomly sub-sampled initial region R_j and then iteratively optimized by EM. The final support set P_j will therefore differ from the initial R_j that gave rise to the hypothesis. Component explanation measures which portion of the original component R_j remains assigned at the end:

$$C_{xpl} := \frac{|P_j \cap R_j|}{|R_j|}, \quad C_{uup} = 1 - C_{xpl} \tag{5.69}$$

A low value of C_{xpl} indicates that K_j explains only part of the point set R_j. A high *unused potential* C_{uup} suggests trying a different hypothesis generating algorithm. For example, the most efficient algorithms in section 5.4 building *straight-axis* hypotheses are tried first and by default. If the initial region actually comes from a complex GC with a curved or branched axis, the algorithm will recursively split the region R_j, but may do so in a sub-optimal way. Upon termination many insignificant 'cleavage products' with unassigned points may remain. A high unused potential C_{uup} then suggests a second trial: calculating a dense foot point cloud and extracting the axis using the slower principal curve algorithm from section 5.3.4.

Visibility sector, Viewing angles: Not only the intrinsic object parameters, also some view-dependent parameters are useful: for rendering, for estimating the SoR surface area and its point coverage, and for assessing the likelihood of relative poses estimated between different views by the SLAM algorithm [Koh07]. The *visible sector* interval $[\omega_-, \omega_+]$ indicates smallest respectively largest angles of rotation about the SoR axis for any partially visible meridian. Rotation angles refer to some unit vector $\mathbf{q}$ in the cross section plane orthogonal to the *axis direction* and to the *mean normal vector* $\bar{\mathbf{n}}$ of the support set. The meridian angles ω_i are computed by formula (5.70); a geometric interpretation is given in figure 5-26: the sinus of the meridian angle of any point is the ratio between its length of projection onto vector $\mathbf{q}$ (numerator) and the point distance from the SoR axis (denominator). For a point located in the direction of the mean normal $\bar{\mathbf{n}}$, $\omega = 0$ [rad].

$$\sigma_i := \frac{\Delta \mathbf{p}_i^T \cdot \mathbf{q}}{d(\mathbf{p}_i, L(\mathbf{s}, \mathbf{a}))} = \sin \omega_i \quad where \quad \mathbf{q} := \frac{\mathbf{a} \times \bar{\mathbf{n}}}{\|\mathbf{a} \times \bar{\mathbf{n}}\|} \quad \Rightarrow$$
$$\omega_- = \sin^{-1}\left(\min_{i \in P_j} \sigma_i \right) \in \left[-\frac{\pi}{2}, 0 \right], \quad \omega_+ = \sin^{-1}\left(\max_{i \in P_j} \sigma_i \right) \in \left[0, \frac{\pi}{2} \right] \tag{5.70}$$

Let $\mathbf{p}_-$, $\mathbf{p}_+$ denote surface points including the extreme angles ω_- resp. ω_+. With the according ray directions $\mathbf{r}_-$, $\mathbf{r}_+$ pointing *orthogonally from* the axis *towards* $\mathbf{p}_-$, $\mathbf{p}_+$ and the known projection and radius bounds $r(s_{min})$, $r(s_{max})$, two *tangent line segments* delimit the visible sector:

$$\vec{\mathbf{t}}_\pm \;:\; LineSegment\left(\mathbf{t}_{A,\pm}, \mathbf{t}_{E,\pm}\right) \quad with \quad points$$
$$\mathbf{t}_{\{A|E\},\pm} := \mathbf{s} + s_{\{min|max\}} \cdot \mathbf{a} + r\left(s_{\{min|max\}}\right) \cdot \frac{\mathbf{p}_\perp\left(\mathbf{p}_\pm, L(\mathbf{s},\mathbf{a})\right)}{\left\|\mathbf{p}_\perp\left(\mathbf{p}_\pm, L(\mathbf{s},\mathbf{a})\right)\right\|} \tag{5.71}$$

When no occlusion by other parts occurs, the two tangents correspond to the *limb* segments mentioned in the GC/SoR literature [ZeN96]. Casting shortest viewing rays $\mathbf{e}_\pm$ *from* each tangent line segment $\mathbf{t}_\pm$ *to* the sensor origin and observing their angles included with the normals $\mathbf{n}_\pm$ at the tangents obtains two extreme *viewing angles* $\gamma_\pm$ onto the SoR surface. A mean *viewing angle* γ between the mean normal $\bar{\mathbf{n}}$ and the center viewing ray $-\mathbf{c}$ is also defined. Figure 5-26 graphically explains the variables used to calculate the visibility information.

$$\mathbf{e}_\pm := l \cdot \left(\mathbf{t}_{E,\pm} - \mathbf{t}_{A,\pm}\right) - \mathbf{t}_{A,\pm} \quad where \quad l := \max\left(0, \min\left(1, \mathbf{t}_{A,\pm}^T \cdot \left(\mathbf{t}_{E,\pm} - \mathbf{t}_{A,\pm}\right)\right)\right)$$
$$\mathbf{n}_\pm := \begin{cases} Rot\left(\mathbf{a}, \omega_\pm\right) \cdot \bar{\mathbf{n}} & if \quad SoR \quad convex \\ Rot\left(\mathbf{a}, -\omega_\mp\right) \cdot \bar{\mathbf{n}} & else \end{cases} \tag{5.72}$$
$$\gamma_\pm := \angle\left(\mathbf{e}_\pm, \mathbf{n}_\pm\right) \overset{!}{\in} [0, \pi/2] \qquad \gamma := \angle\left(-\mathbf{c}, \bar{\mathbf{n}}\right)$$

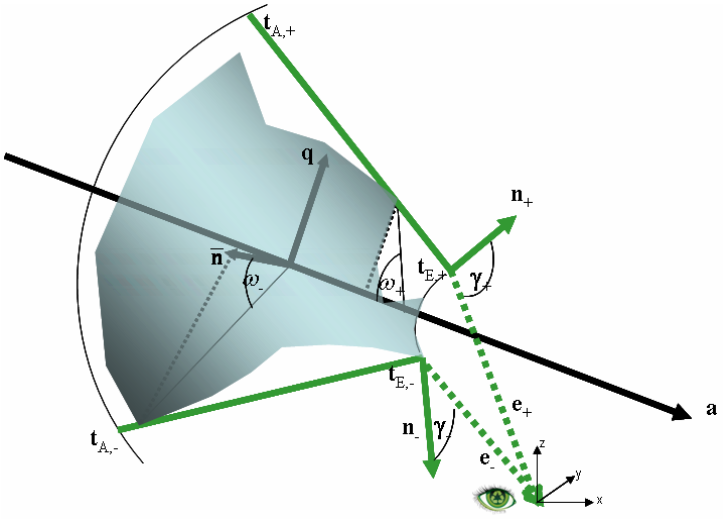

fig. 5-26 Illustration of the visible SoR sector; see text for explanations

The angle information γ, γ_-, γ_+ is transformed into a confidence value $C_{eye} \in [0,1]$ coding our expectations about a true positive hypothesis:

- Any visible feature must have viewing angles below $\pi/2$, i.e. the surface normal and the viewing direction include a positive dot product. Using (5.70) in practice somewhat overestimates the true meridian angles ω_-, ω_+ as every single point contributes. Still, the viewing angles γ_-, γ_+ should not much exceed $\pi/2$ for a true SoR.

- On the other hand, γ_-, γ_+ should come close to the 90° viewing angle of a self-occluding boundary, *unless* some closer surface occludes the tangents. A too small viewing angle

onto a SoR border raises the question as to why is not seen more of it. The likely answer is: a *rounded crease* edge joining two patches but no genuine SoR sector is observed.

- The magnitude of the visible angle sector $(\omega_+ - \omega_-)/\pi$ (π being the maximum for a point-like sensor origin) might itself be used as a confidence measure. Theoretically, this is justified by the sensitivity of parameter estimation which sharply increases as the captured *arc angle* decreases. This has been shown for 2D circle fitting by expressing the Cramér-Rao bounds as a function of the arc angle [ZeC04] and principally holds in 3D for SoR axis estimation more than ever. Rabbani [Rab06] for example demands a visible angle sector of ≥90° ($\pi/2$) for cylinders which in our THERESA plant is often not obtained. We do not impose such a condition because it would raise our number of missed (false negative) SoR objects too far.

Surface coverage: The total patch area A_{Patch} covered by all points assigned to a SoR is related to its model surface area A_{Sect}; their quotient gives the *model surface coverage* SC. The SoR area A_{Sect} is the length of the profile curve times the visible angle sector $[\omega_-, \omega_+]$. For a piecewise linear radius function with M segments as profile curve, the length integral is approximated by the following sum using the notation from section 3.1.

$$SC := \frac{A_{Patch}}{A_{Sect}} \quad where$$
$$A_{Sect} = (\omega_+ - \omega_-) \cdot \int_{s_{min}}^{s_{max}} r(s) \cdot \sqrt{1 + \dot{r}(s)^2}\, ds \approx (\omega_+ - \omega_-) \cdot \frac{(s_{max} - s_{min})}{M} \cdot \sum_{j=0}^{M-1} \frac{r_j}{\cos \delta_j} \tag{5.73}$$

A low SC value, i.e. a small data area supporting a large model surface indicates a false positive hypothesis. Surface coverage is complementary to component explanation.

Tangential tests: These tests check if a SoR region is better explained by a planar patch or by *two* or *three* adjacent ones forming a crease or a corner such that those patches are more or less co-tangential to the original SoR. A greedy nearest-neighbor classification into up to three plane hypotheses is first performed on the SoR support set. For testing the degree of planarity, the PCA matrices of the point and the normal vectors are formed, yielding three point eigenvalues $\lambda^{(\mathbf{p})}$ and three normal eigenvalues $\lambda^{(\mathbf{n})}$. In case of a planar patch, the point tensor [TaM99] is 'plate-like', the smallest eigenvalue being clearly separated from the two larger ones, whereas the normal tensor is 'stick-like', the largest eigenvalue being well separated from the two smaller ones. The smaller the combined eigenvalue ratio

$$EWR := \frac{\lambda^{(\mathbf{p})}_{\min}}{\lambda^{(\mathbf{p})}_{mid}} \cdot \frac{\lambda^{(\mathbf{n})}_{mid}}{\lambda^{(\mathbf{n})}_{\max}} \in [0,1] \tag{5.74}$$

the less likely it is that the points come from a true SoR. To check the degree of *tangentiality* of an oriented flat point ($\mathbf{p}$, $\mathbf{n}$) with respect to a rotation surface, its orthogonal projection onto the SoR axis $\mathbf{p}_\perp(\mathbf{p}, L)$ is projected once again in the normal direction $\mathbf{n}$. The length of this projection vector is related to the radius near $\mathbf{p}$ and compared to the value ± 1 for an ideal tangent, to yield a tangent error ε_T. This test is applied to the center $\mathbf{c}_k$ and normal vector $\mathbf{n}_k$ of

each alternative flat patch k formed. The larger the tangent error ε_T and the larger the eigenvalue ratio (5.74), the higher is our confidence C_{SoR} in the hypothesis being a true SoR:

$$\begin{aligned} \varepsilon_T(\mathbf{c},\mathbf{n}) &= \min\left(\left|\frac{\mathbf{n}^T\cdot\mathbf{p}_\perp(\mathbf{c},L)}{r(s)}\mp 1\right|,1\right) && (\text{'}-\text{'}: convex,\quad \text{'}+\text{'}: concave \quad SoR) \\ C_{SoR} &= \sum_{k\le 3}\varepsilon_T(\mathbf{c}_k,\mathbf{n}_k)\cdot EWR_k \in [0,1] && (k: \quad index\ of\ a\ planar\ patch) \end{aligned} \tag{5.75}$$

Final weight: The final weight W of a SoR hypothesis combines the original weight w with the viewing confidence C_{eye}, the model surface coverage SC, and the tangent criterion value C_{SoR} as follows:

$$W = w\cdot C_{eye}\cdot SC\cdot C_{SoR} \tag{5.76}$$

A product (conjunction) of different confidence values instead of a weighted sum has been used in (5.76) because the terms, though normalized to [0, 1] are not directly comparable in magnitude. Therefore, their relative weights are unknown. The absolute values of W are unimportant anyway, they only serve as an ordering criterion. A certain percentage of the sum of weights is used as a cut-off value for hypotheses.

An example of hypothesis evaluation is shown in figure 5-27. On the left is shown the state after parameter optimization. There are several conflicting object hypotheses around the tank Kessel_2012, and, to a lesser extent, around the small pump unit to its left, spawned by separate image regions but competing for the same space. Also, false-positive cylinder hypotheses appear near the roof edges formed by the planar surfaces of the ground platform. After hypotheses evaluation, most false objects have disappeared as shown on the right. The decrease in confidence by hypotheses testing according to (5.76) is graphically indicated by the higher degree of transparency; even the viable hypotheses shade off to some extent.

Hypothesis pruning will however not always be fully effective which will be investigated in further experiments in chapter 6.

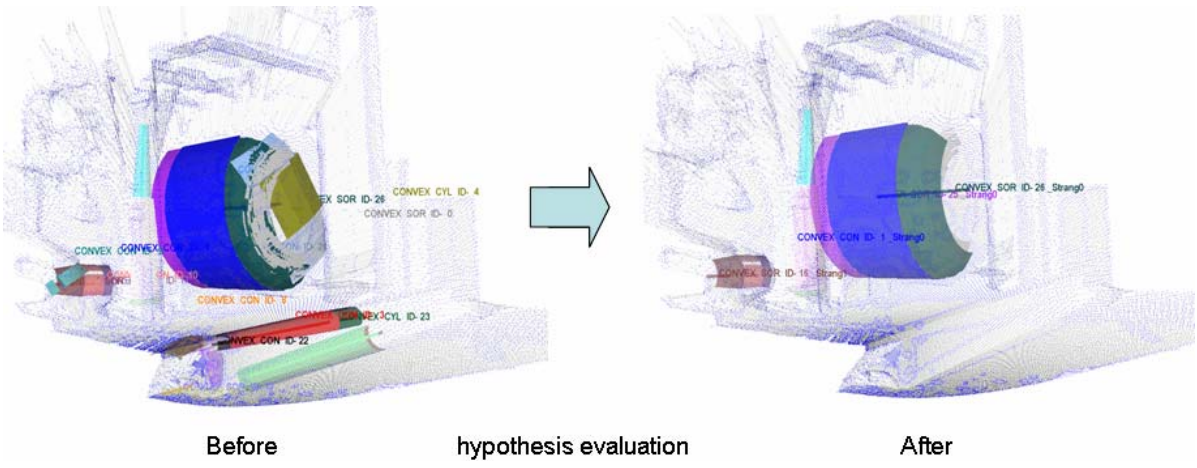

fig. 5-27 Hypothesis evaluation of the range view Kessel_2012.vw1

5.7 SoR relations

An important final step towards an attributed feature graph is to classify spatial relations among cone/SoR segments and to group the parts into coherent units, such as flanged pipe ducts. The relation structure is essential to narrowing down the surface correspondence probabilities between different views, for estimating the motion and for closing loops. As soon as the observer motion becomes known, the relations are extended to span objects from different views. Thus, the primitives are grouped into increasingly larger units. Estimating relations is part of hypothesis evaluation and itself contributes to the detection of more false-positive hypotheses.

Rabbani [Rab06][Rab07] recognizes complex and powerful though special configurations that became design standards in modern petrochemical plants, for example *T-junctions* of flanged pipes. Compared to his work we model in a first step rather weak and general binary relations which may apply to any pair of truncated cone, cylinder, or SoR elements, i.e. require no design assumptions, but are not very distinctive and informative. By grouping weak relations of this kind in a bottom-up fashion we can still derive more specific and meaningful (n-ary) relationships. The following basic relation types are distinguished.

Conflict: Two distinct hypotheses of SoR elements whose pose and shape parameters imply mutually penetrating objects; at most one hypothesis may consistently explain the surface geometry.

Redundant: A special *conflict* relation where both elements have quite similar parameters; one is sufficient to explain the geometry.

Coaxial: Two spatially disjoint elements are approximately aligned on a single straight axis line.

Coplanar: Two elements approximately lie in a common plane spanned by their axis directions, i.e. the two infinite axis lines intersect when widened by suitable small radii, but the bounded SoR volumes do not intersect physically.

Junction / Branch: A special case of coplanar axes where both SoR elements physically intersect in a particular way: exactly one end is contained in ('runs into') the other SoR element. Topologically, a configuration of two SoR elements with three end points is formed.

Parallel: Both elements have very similar axis directions but they are neither coaxial nor do their bounded volumes overlap.

Other: Various kinds of relations known from the specific hypothesis generating procedures, not necessarily geometric. The strongest one is the *Parent*/*Child* relation which implies also being *coaxial* and topologically *connected*: the parent is a cone and its children are SoR extensions at its left or right end. The relation *Smoothly Connected* holds if several elements are *siblings* of a single hypothesis, i.e. stem from a single smooth region which has been decomposed using one of the algorithms from sections 5.3.4 or 5.4.5. A still weaker hypothesis is *Adjacency*: the elements were generated from *topologically adjacent* seed regions which by itself permits no conclusion such as being contiguous.

Geometric relationships of the kind *Coaxial, Coplanar,* or *Parallel* alone tell nothing about connectedness, i.e. affiliation to a single connected pipe duct. If, in addition, the relation *Connected* holds, the resulting combinations *Coaxial-Connected, Coplanar-Connected, Parallel-Connected* are more distinctive. Connectedness implies affiliation to a single duct, because the SoR elements either came from the same connected component (*Smoothly Connected*), or it was explicitly verified that their *volumes* adjoin or overlap but without a *Conflict* relation being present.

Depending on the classification results, different actions on the set of SoR hypotheses are performed. A *Conflict* relation causes the weaker one of two hypotheses (with lower quality) to be deleted. No fusion of object parameters is attempted in this case. In the similar case *Redundant*, a merged parameter vector is created from two similar hypotheses.

Binary *Connectedness* relations initiate a labeling process called duct linking. Every truncated cone or SoR element has two end points. A connection relation binds two, in case of the special *Junction* relation only one, end point(s) from different elements, and creates a partial ordering on the set of SoR elements or segments. In this way, a spanning tree of relational edges is constructed. The remaining unbound (free) endpoints in this tree indicate entry/exit points from ducts.

The *Coaxial* relation implying that several SoR segments have a common axis line plays a unique role. It initiates a loop back from the verification to the hypothesis generation stage. This time, the union of point sets from originally independent parts is treated jointly, as a SoR group hypothesis. Axis estimation, using for example the Plücker approach in section 5.4.5, is repeated and refined. This opens a chance to obtain a geometrically more faithful estimate of the shared axis direction, regardless of the affiliation of the parts to pipe ducts.

Most relations, in particular the distinctive parallel and coplanar ones, yield pose-invariant constraints for the SoR feature association in the SLAM process.

Figure 5-28 illustrates by means of a coarse flowchart or decision tree how these kinds of relations are determined. Actually, no crisp Boolean decisions are made in the individual decision nodes. Each decision is mapped to a fuzzy criterion value [0,1], i.e. a degree of fulfillment depending on the attribute values of the SoR parts. Paths taking several decisions sequentially from top to bottom are assigned the product of the respective values. The leaf relation whose path has the maximum degree of fulfillment is chosen as the result.

Some decision criteria in figure 5-28 are fairly self-explanatory, such as similarity of axis directions or mutual relative overlap of the oriented bounding boxes (abbreviated by OBB and oriented with respect to the axis direction). Some criteria are briefly explained:

- Co-planarity criterion: the two axis directions are dissimilar, spanning a plane. The lines' orthogonal distance is small compared to the SoR radii involved. In this sense, the axis lines have a virtual intersection point in 3D.

- Inclusion-Intersect criterion: the conditions of being co-planar are met, and the intersection point lies inside the projected line segment for one SoR, but outside of the other SoR

line segment (the 'second SoR runs into and ends in the first SoR). Exactly one end point of the second SoR axis segment lies inside the volume of the first SoR.

- Redundancy criterion: the similarity of the SoR must be high with regard to both pose parameters (axis lines, projection intervals) and shape parameters (radius functions).

- Coaxial criterion: a 3D line *joining* both SoR *center points* is used as a reference. All axis segment endpoints have a small orthogonal distance to this line compared to the radius magnitudes, and include an acute angle with the joining line.

- Mutual Projection criterion: assuming that the two SoR elements have similar axis orientations and occupy disjoint volumes, i.e. are parallel, their axis segments mutually project onto each other, yielding projection intervals of large mutual overlap.

- Disjoint criterion: assuming that the two SoR elements are coaxial, their axis segments occupy *disjoint* intervals on the joint axis line except for the case that one element has convex and its counterpart concave curvature. This rather rare case of being coaxial and overlapping yet separate and without conflict occurs if both the outer wall of some pipe and the outer or inner wall of the same or an enclosed pipe are simultaneously visible.

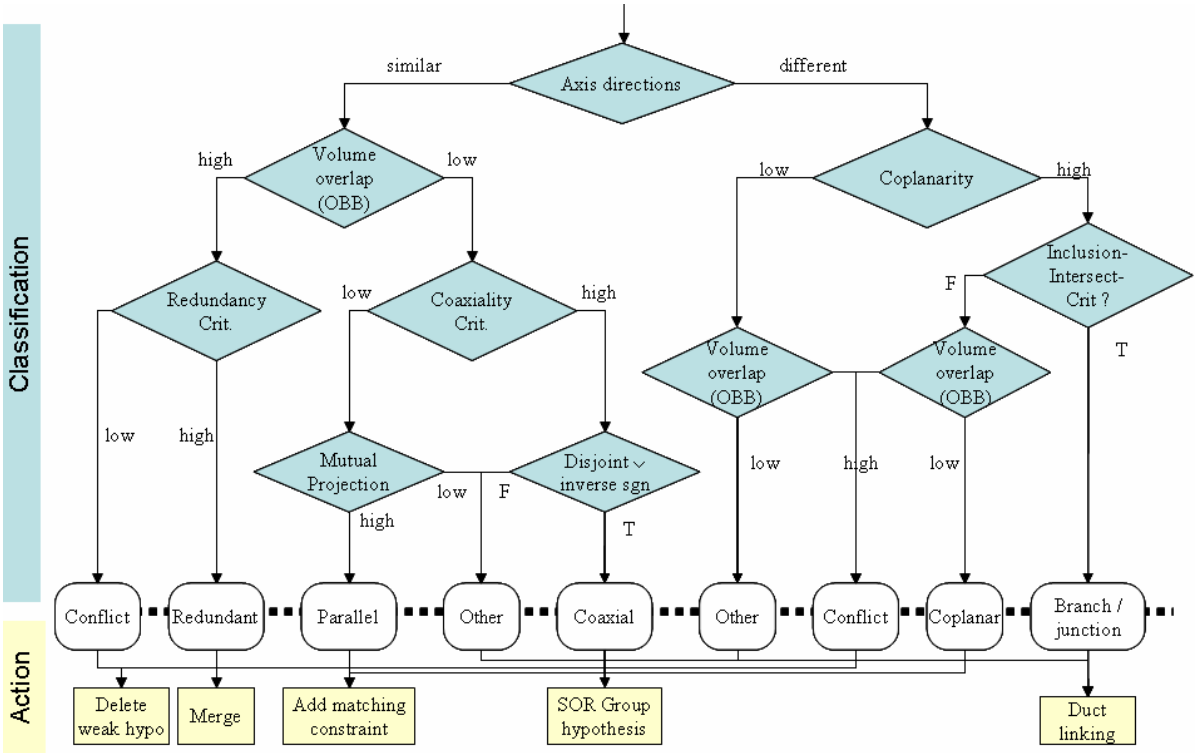

fig. 5-28 Decision tree for determining and classifying the relations between bounded SoR elements. On the bottom, some resulting actions are indicated.

The following two figures 5-29 and 5-30 illustrate two example range views with the relations and ducts estimated, and the tables 5-1 and 5-2 show the corresponding relation tables generated.

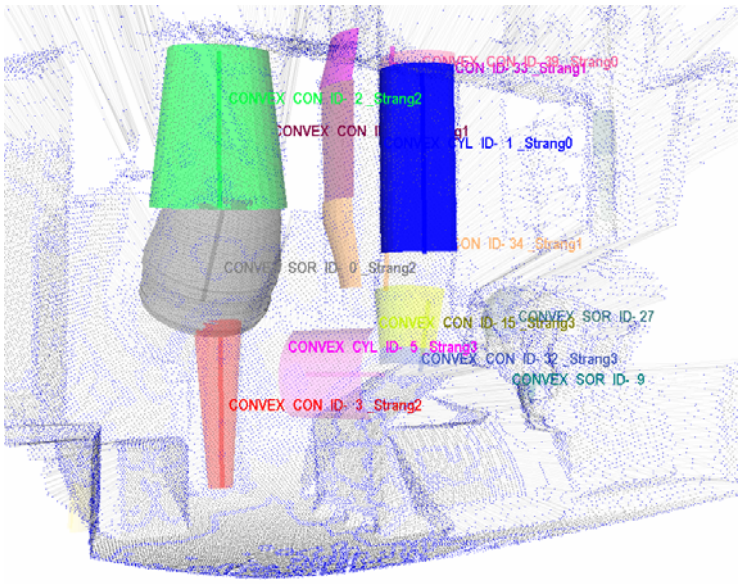

fig. 5-29 SoR elements with relations for range view Baugruppe_6201_vw0

Duct		0	1	2	2	1	3	1	0	3	2	3
	Sor-ID	1	34	2	0	33	5	13	39	15	3	32
0	1		||	||			| _	||	Parent	_ _		_ _
1	34	||				Siblg		Conn				
2	2	||			Conn			||			_ _	
2	0			Conn							_|_	
1	33		Siblg					Child	||	_ _		_ _
3	5	| _							| _	Siblg		Siblg
1	13	||	Conn	||		Parent						
0	39	Child				||	| _			_ _		_ _
3	15	_ _				_ _	Siblg		_ _			Conn
2	3			_ _	_|_							
3	32	_ _				_ _	Siblg		_ _	Conn		

Legend

||	parallel	Parent,	The child is a left or right SoR
_ _	coaxial	Child	extension of its parent cone
| _	coplanar	Siblg	SoR are off-springs (siblings) of
_|_	branch		the same region

Conn Connected - volume overlap or hypotheses spawned by adjacent regions

Table 5-1: Relations generated for range view Baugruppe_6201_vw0

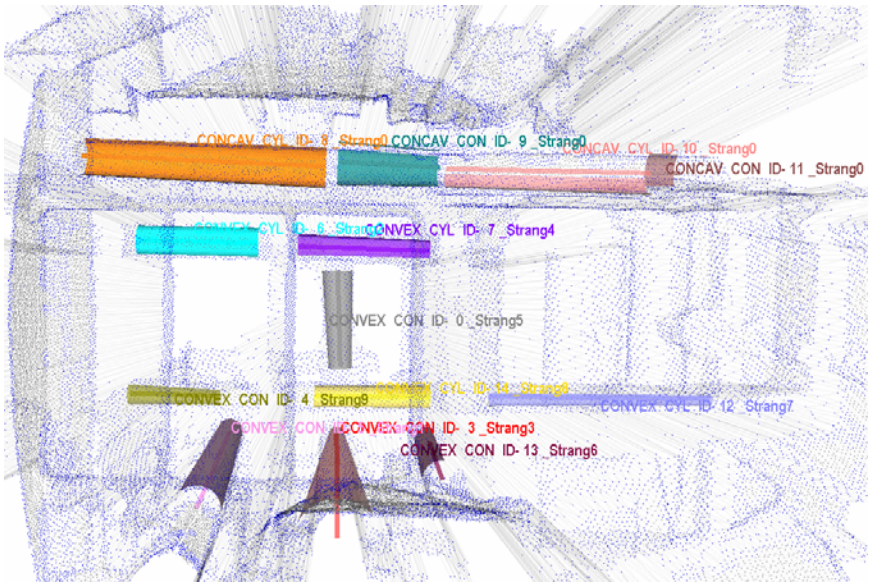

fig. 5-30 SoR elements with relations for range view Wasserleit_B2005_vw0

Duct		0	1	2	3	4	5	6	7	0	8	0	0	9
	Sor-ID	8	5	6	3	7	0	13	12	9	14	11	10	4
0	8			||						Conn		_ _	_ _	||
1	5				||			||						
2	6	||				_ _								||
3	3		||				| _	||						
4	7			_ _						||	||			
5	0				| _									
6	13		||		||									
7	12										_ _	||	||	_ _
0	9	Conn				||					||	_ _	Conn	
8	14					||			_ _	||				_ _
0	11	_ _							||	_ _			Conn	
0	10	_ _							||	Conn		Conn		
9	4	||		||					_ _		_ _			

Table 5-2: Relations generated for range view Wasserleit_B2005_vw0. Duct 0 contains several coaxially-connected *concave* cones and is probably part of the support structure (U profile) but is wrongly characterized as a pipe duct.

5.8 Algorithm integration

Extracting rotation symmetric objects from range views in process plants and estimating their parameters in real time requires many different tools some of which have been presented. This section integrates these methods into overall modules which on their turn fit into a feature based 3D mapping system. The mapping module should implement a good compromise between the conflicting goals of estimation and timing performance. Regarding estimation performance, the focus is on false SoR objects detected and true SoR objects undetected (false positive / false negative hypotheses), as well as on the stability and the accuracy of the model parameters under input perturbations. Real-time performance means creating useful SLAM features on-the-fly, keeping up with data capture. From the empirical investigation (chapter 6) a standard set of methods and an overall control structure emerged which in many cases yielded a good performance.

This scheme is illustrated by the control and data flow diagram (figure 5-31) and will be explained. As explained before, boxes denote processing instances (activity), rounded rectangles or ovals denote data objects. Ovals in bold lines near a module boundary indicate externally visible interface data. While solid arrows symbolize data flow between activity and data (reading or writing), dotted arrows indicate flow of control between activities. The module in figure 5-31 expects as input ordered range images with oriented 3D points and curvature features, optionally a region of interest. It runs **three stages** distinguished by respective colors in figure 5-31.

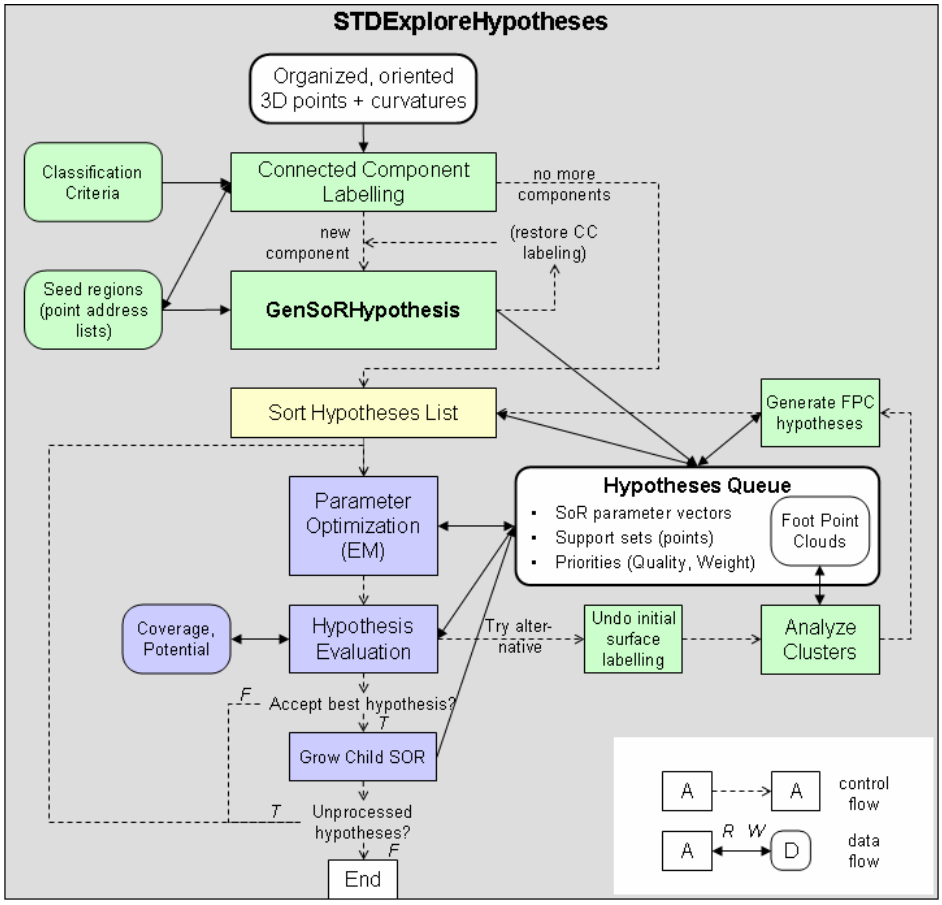

fig. 5-31 Data and control flow of the standard algorithm (STD) for SoR extraction

1. Partitioning into connected components of homogeneous curvature type. Region homogeneity, shape, and sizes are controlled by user-selected criteria for classification and labeling, including conditional morphology (section 5.1). On a subset of the regions, SoR hypotheses are created (*GenSoRHypothesis*) using methods from sections 5.2 - 5.4.

2. Building a hypothesis priority queue sorted by decreasing importance which is preliminary information from the first stage. A hypothesis comprises an initial parameter vector, a support set of points, and quality and importance (weight) values.

3. Sequentially processing and emptying the queue, performing parameter optimization and hypothesis evaluation (sections 5.5, 5.6). Hypotheses may fail and be discarded; new ones are generated by trying different methods on the respective region and are inserted back into the priority queue. Optimization is repeated until the queue becomes empty.

The initial priority ordering at stage 2 does have an influence on the final result because hypotheses created from disjoint seed regions compete for common data points in the E-step in stage 3, but do so in a preemptive fashion and not independently. Weaker hypotheses tend to be further penalized as their final weight depends on the amount of conflict encountered during their growth. Focusing on 'big' hypotheses is a strategic decision, a compromise not reconcilable with pure probabilistic reasoning (section 5.5.1).

The standard method for hypothesis generation *GenSORHypothesis* uses closed-form PCA-like algorithms or direct optimization of gridded parameter values and performs no heuristic search. On images showing SoR with linear axis and linear radius function it achieves a good cost-benefit ratio. In some cases, however, other methods may be needed, in particular the principal curve analysis on foot point clouds. Those cases are only recognized after hypothesis evaluation: mostly, a large fraction of the seed region remains unassigned, or, conversely, only a small part of the model surface is explained by data points. In other words, the surface coverage SC (5.73) or component explanation C_{xpl} (5.68) figures are low and the unused potential C_{uup} high. In such cases, the alternative method of *principal curve analysis* will be executed on the foot point cloud (FPC) retained from the previous hypothesis generation; there is no need to run the entire foot point transformation on the image. After undoing the current surface labeling, the FPC is decomposed using algorithm *Analyze* from section 5.3.4 which generates SoR hypotheses in the same format as *GenSORHypothesis*. After optimization and comparison with the first result, the hypotheses tallying most weight on that region will be established if the weight is sufficient for acceptance.

5.8.1 Standard hypothesis generation

The internal working of algorithm *GenSORHypothesis* is illustrated on the left of figure 5-32 and explained next. This algorithm is applied only to regions where minimum and maximum curvatures essentially differ in magnitude, i.e. to ridges, valleys, and possibly saddle ridges and saddle valleys, but no planar, spherical (pit, peak) or saddle surfaces of minimal type.

At first, a subset of $N<N_0$ points is selected by uniform *random sub-sampling*. The subset size ranges from $S_{min}\approx 100$ for small regions, sampling a large fraction $fr_{max}\approx 0.95$, to $S_{max}\approx 1500$ for large regions, sampling a small fraction $fr_{min}\approx 0.2$, according to the formula:

$$N = \frac{S_{max} S_{min} \left(fr_{max} - fr_{min} \right) + N_0 \cdot fr_{max} fr_{min} \left(S_{max} - S_{min} \right)}{fr_{max} S_{max} - fr_{min} S_{min}} \tag{5.77}$$

Apart from efficiency concerns - bounding the overhead for large components - the main reason for sub-sampling is to obtain empirical covariance data for the parameter estimation in a rather 'cheap' way, see section 6.2 on stability assessment.

The next step on the sub-sampled points, *InitPCA*, determines the sign of the Mean curvature sum and decides if the resulting surface type (convex or concave) is consistent with the region curvature type or shape index. A coarse radius of curvature $r^{(0)}$ is also estimated. Principal component analysis (PCA) is then executed:

a) on the point normals, to assemble from the eigenvector directions an initial coordinate frame containing tangent, normal, and cross section directions (section 5.4.1),

b) on the normal ray linear complexes (section 5.4.4, the 6D and 7D cases treated jointly). A sorted candidate list of 3D axis directions is assembled. From both the 6x6 and the 7x7 matrices, $i \leq 3$ smallest eigenvalues are selected with a cut-off index i, $\lambda_{i-1}/\lambda_i < \lambda_i/\lambda_{i+1}$, so as to keep together multiple eigenvalues. The most closely corresponding 3D directions obtained rank highest in the axis candidates list.

The initial tangent direction $\mathbf{t}$ from step a) is among the valid candidates for the axis direction. All candidates are tested sequentially as straight-axis hypotheses (*EstimateAxis*). An axis center point $\mathbf{s}$ is associated with each direction (eq. 5.50), and the degree of rotational symmetry ($0 \leq Q \leq 1$) is assessed by combining the radial symmetry and the properties of the NPA angle distribution as described in section 5.4.5. As soon as a candidate passes the rotational symmetry test, an SoR hypothesis is generated (*SetupSORHypo*). If all tests fail, the set of surface points is split and *GenSORHypothesis* is invoked recursively on the left and right point subsets. For splitting (*SplitSORHypothesis*), point projection onto the tangent direction $\mathbf{t}$ serves as the ordering criterion, and the projection *bin* maximizing some distance criterion is chosen to split the point set as explained in section 5.4.5.

If all axis candidates failed but the point set can be classified as a *small* component already, further splitting will not succeed and the directional constrained foot point transformation (DC-FPT, section 5.3.3) is tried as a last resort. Most likely, a badly resolved component of mixed curvature types is the cause. On such components, only, DC-FPT showed a superior performance to the direct methods. For ordering the cross sections, DC-FPT uses an optimized version of the initial tangent $\mathbf{t}$, which is rotated about the component normal until the radius function becomes most linear (algorithm *FindInitialTangent* from section 5.4.2).

The DC-FPT internal data flow is refined on the right of figure 5-32. At the first stage, the point set is divided into cross sections, i.e. narrow intervals of projection onto $\mathbf{t}$. For each bin h, the radius r_h minimizing the spread is found (C-FPT). At the second stage, the radii distribution is built and outlier values are pruned. From the inliers radii, foot points $\{\mathbf{f}_h = \mathbf{cp}_h + r_h \cdot \mathbf{cn}_h\}$ are calculated. The axis $L(\mathbf{s}, \mathbf{a})$ is estimated as a regression line through the foot points. Its spread, measured by the magnitude of the smaller eigenvalue, must be smaller than the average within-bin spread of foot points to accept the straight-axis hypothesis. The relative difference of these two values determines the hypothesis quality in this special case ($0 \leq Q \leq 1$), which is more tolerant than the degree of rotational symmetry.

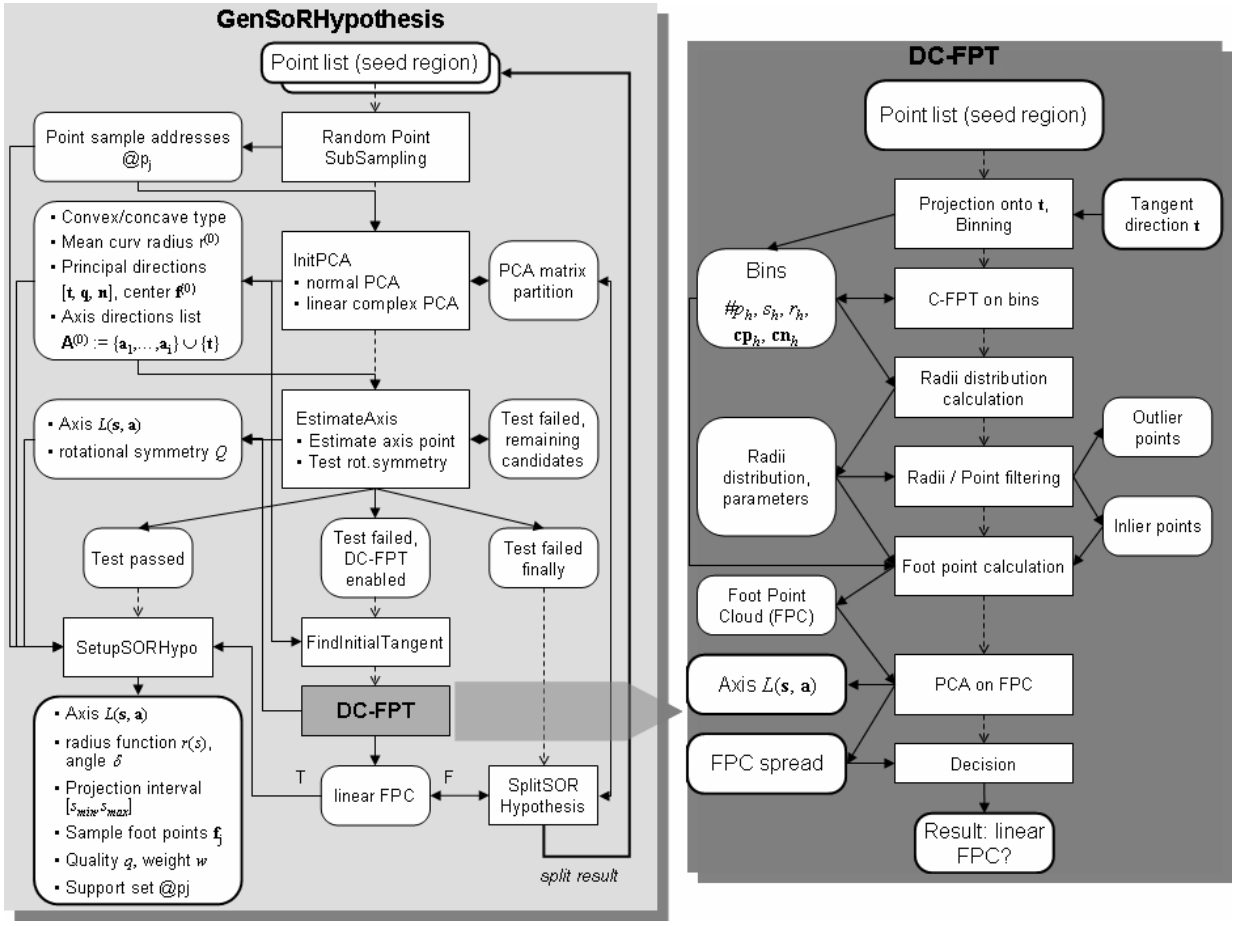

fig. 5-32 Data and control flow of the standard algorithm GenSORHypothesis (left). On the right is shown in detail the internal algorithm DC-FPT which is invoked if the PCA-based method fails to produce a SoR hypothesis.

Returning to the main algorithm *GenSORHypothesis*, the method *SetupSORHypo* is called after passing the linear-axis hypothesis test using the internal quality Q as a score value. The radius function $r(s)$ is estimated next (section 5.2.4, *UpdateProjRadius*). Quality q, weight w, and the sub-sampled point set are added to the hypothesis. A set of foot points $\{\mathbf{f}_i\}_{i=1\ldots N}$ is also generated using the positive radius function: $\mathbf{f}_i = \mathbf{p}_i \pm r(s(\mathbf{p}_i))\cdot\mathbf{n}_i$ where '+' sign applies to a *concave* and '-' to a *convex* SoR. Foot points are retained for later principal curve analysis, in case that the hypothesis should fail to produce a valid EM optimization result.

5.8.2 FPT algorithm

The FPT algorithm is an alternative to the standard (STD) algorithm, combining elements of the CHAIN algorithm (section 5.2) with the directional constrained foot point transformation (DC-FPT, section 5.3.2). The resulting foot point cloud is decomposed using the principal curve algorithm from section 5.3.4. Figure 5-33 illustrates the data flow.

Randomly sampled points from the input component are used as seed points for exploring MIN and MAX chains in forward and backward directions (*FindFootPt_Chain*). The purpose of these chains here is to create *local* cross sections for the FPT to work with. 'Local' means: each MIN/MAX *chain* pair creates its own Darboux frame and indicates an associated cross

section direction, and not the entire *component* tangent direction is imposed to project and group the points into cross sections. The chain frame is located in the chain midpoint which is a true surface point but in general not the seed point from which the search started.

For the FPT application, a few detail improvements were made on the chain search from section 5.2.2: principal curvature directions at each point are better resolved by looking at least two points ahead. Candidate points must satisfy stricter conditions on their curvature type to be included in the chain, based on a localized curvature distribution.

After calculating a local frame and a foot point for each chain, the obtained foot point cloud is decomposed using algorithm *Analyze* from section 5.3.4. A component frame assembled from the chain normal directions supplies several directions for the principal curve algorithm to start with. Rated SoR hypotheses are generated from the terminal leaves in the same format as by *GenSORHypothesis*. The remainder of the algorithm (hypothesis optimization and evaluation) works as for the standard algorithm.

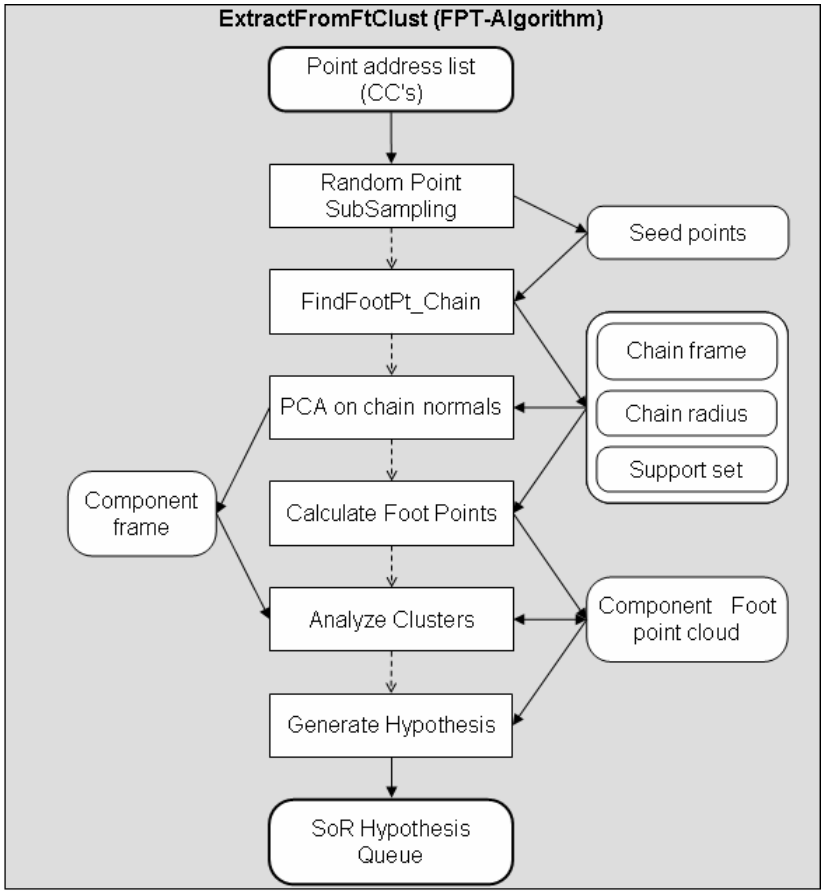

fig. 5-33 Data and control flow of the foot point transformation (FPT) algorithm

5.8.3 Chain algorithm

For hypothesis generation, the Chain algorithm uses the chain search described in section 5.2; its remaining parts - hypothesis optimization using EM and hypothesis evaluation - work as for the standard algorithm. The rather simple data flow is illustrated in figure 5-34.

From a connected component's point address list a specified fraction of points is randomly sub-sampled as seeds for chain search. Searching maximum length chains of MIN and MAX curvature has been described in subsection 5.2.2: after testing the seed point for suitability, the chain search works in forward and in backward direction and builds the chain data structures. While expanding, the links to other chains via shared points are found. A relational chain structure is formed, and linked chains are put into *connected components*. From each set of MIN and MAX chains forming a connected component, a SoR hypothesis is seeded next. The algorithm *Param_EstimateFromChain* explained in subsections 5.2.3 and 5.2.4 estimates the SoR parameters; the reader may refer back to figure 5-10 providing the next level of detail of data flow. Some chains may be considered outliers as to their curvature radii or tangent directions; those chains are not used for the current hypothesis estimation but are 'recycled'. From a collection of discarded MIN/MAX chains new hypotheses may be formed later.

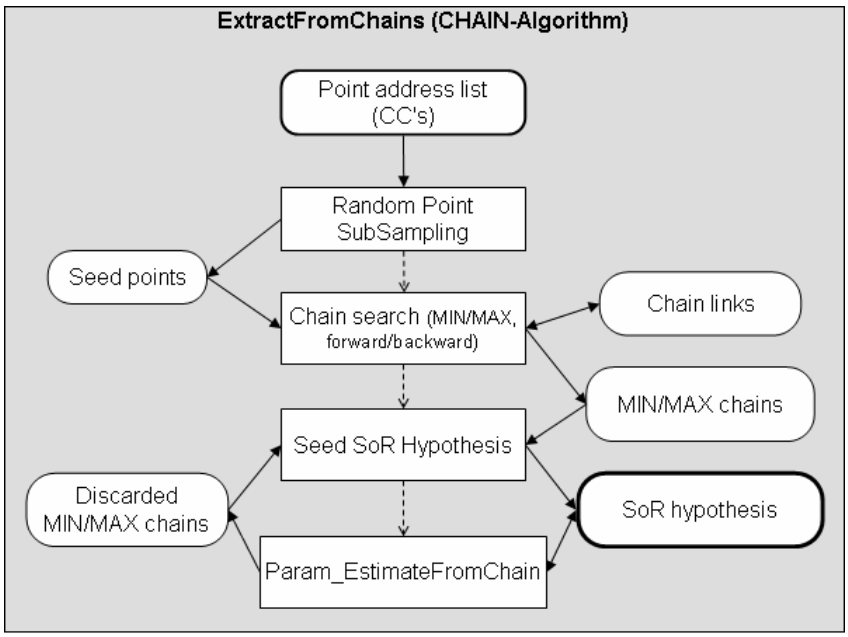

fig. 5-34 Data and control flow of the CHAIN algorithm

6 Experimental results

Our overall design goal is to achieve a best compromise between conflicting goals of estimation and timing performance (reliability, stability, accuracy, real time). From a theoretical point of view we are unable to answer which method, or combination thereof, is best suited to this goal. Experimental comparisons have been conducted on range views from the thermal process plant THERESA (figure 6-1) and some quantitative assessment has been made.

fig. 6-1 Thermal waste processing plant THERESA

6.1 THERESA examples

In this section experimental results are visualized and the performance of three integrated algorithms from section 5.8 - STD, FPT, and Chain - is discussed. The key focus is on the strength of different methods *generating starting values* (SoR hypotheses); for parameter *optimization* all algorithms use the same EM algorithm.

More than 100 scene views were captured using the rotating SICK laser scanner with a sensor view field of 100° and ¼° angle resolution, as explained in section 2.3. All raw images were interpolated to build uniform angle grids of 250x250 points. Only the 36 images taken with the slowest scanner rotation have a sufficiently fine radial resolution and were selected for image analysis. These 36 views have been classified into seven principally different groups where images in each group slightly differ in the viewing distances and directions. The groups are also rated judging from the *difficulties* encountered in testing and training the algorithms on them, not necessarily objective criteria found in the scenes themselves:

I - easy, II - moderate, III - demanding, IV- difficult

The experiments cover *single* views; plant mapping will follow. In fact, from the current image suite it would be difficult to construct a map as it constitutes no contiguous path of exploration. Neither global view pose information is available, nor do common landmark features exist. Each type of scene view is rather unique.

Since our focus is on rotation objects, estimated planar surfaces or regions of clutter found have been omitted from the results. In the near future, another study is planned on automatically learning to classify broader classes of objects including slabs, beams and strutting, railings and runways etc. Only raw and isolated object part models are shown; blending and closing of surfaces or Boolean operations to form valid CAD models is not our current topic. Mathematical libraries such as the open geometry kernel OpenCascade provide functions of this type but applying them requires clearly specified modelling goals and guidelines. Our first steps in linking 3D SLAM maps with CAD models will be updating existing CAD models from measured and matched objects, not automatic creation from scratch.

6.1.1 Assembly unit 6201 - I

The first type of scene shows a detail of the exhaust gas system seen from the ground level: three ducts of pipes varying in shape and diameter (600-1200mm). The left duct being an 'ID fan' has a wide vertical pipe - recognizable from a salient gap in the point cloud in figure 6-2 on the upper left - smoothly bending backward. From it a narrow vertical outlet pipe is branching off. To the right another vertical pipe duct (called a 'bypass') is seen possessing a sharp kink partly occluded by an open cardboard box in the foreground. In the background a third vertical duct of larger diameter with conical ends is barely visible, mostly occluded. This mixed scene is typical of industrial installations as-built and in use, showing also planar surfaces of the floor and from metal casings, slabs from the support structure, and a plastic blanket covering a lumber package. Yet it is the simplest one to process: the objects of interest have clear structures, the data has an adequate spatial resolution, and there is little clutter.

The STD algorithm extracts all relevant parts from the three ducts. It chooses to model the visible curved part of the left duct by two piecewise straight axis segments on both of which a circular radius function is estimated. From the occluded rear duct, the central cylindrical part is shaped first and the two upper and lower conical ends are coaxial extensions. Given the data quality, the accuracy of parameters as to location of axis lines, radii, and opening angles seems satisfactory with two exceptions, the conical modelling of the upper left duct due to the gap in the scanned region, and, secondly, the inaccurate axis and radius of the rear duct from which a narrow angle arc (<30°) is available only.

In contrast to STD, the FPT and Chain methods fail to model the curved duct parts correctly as they deal only with truncated cones, i.e. piecewise linear radius functions. The Chain algorithm, in particular, extracts only four partly incomplete parts out of 11 expected ones.

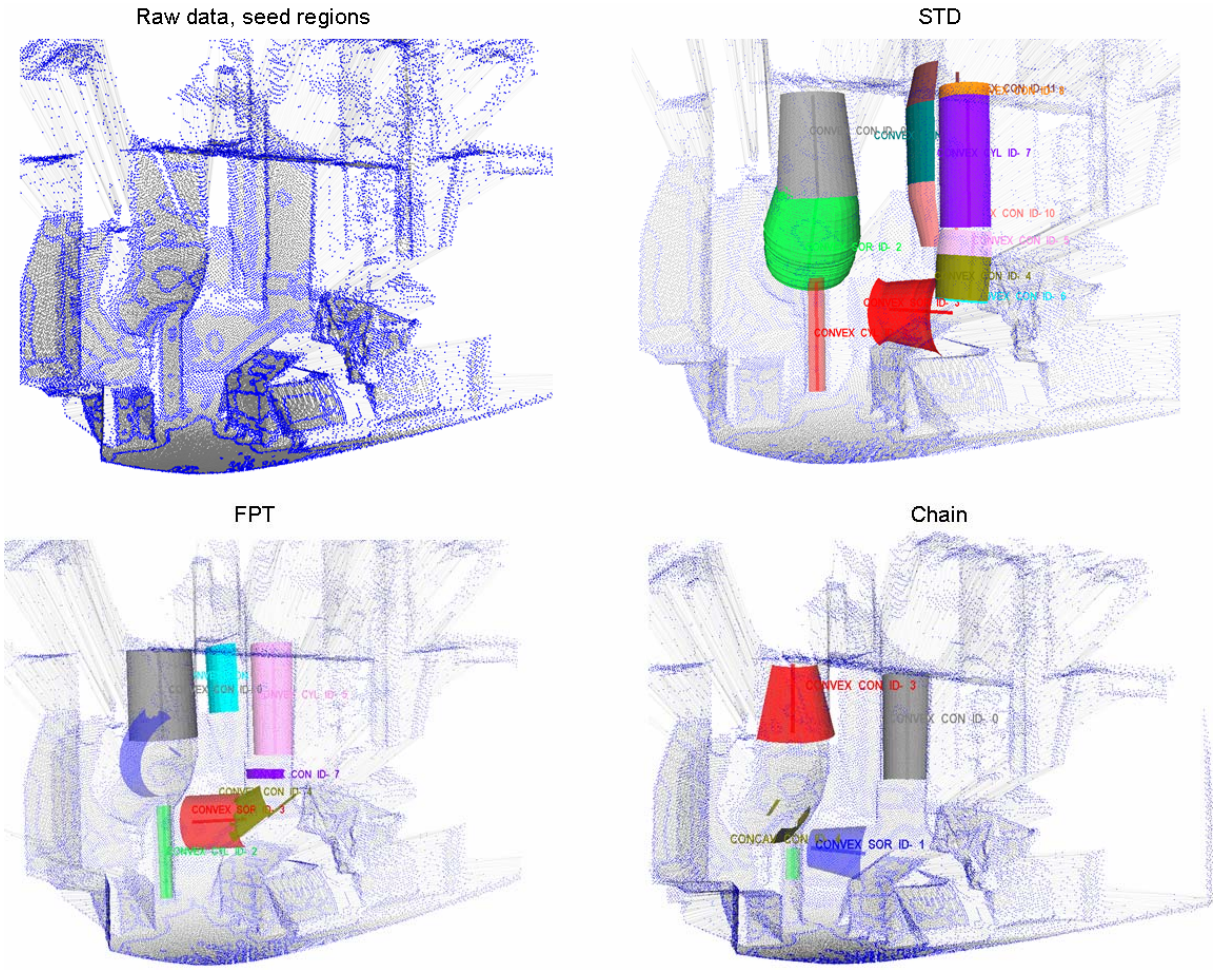

fig. 6-2 Detection and modelling results for scene type Baugruppe_6201

6.1.2 Vessel 2012 - II

This scene view is similar in character but has more planar surfaces due to different ground levels and machine platforms. Only two salient rotationally symmetric objects are visible: in the centre a large horizontal tank of cylindrical shape with spherical/ elliptical closure on the right (≈2000mm diameter); its left side is partly hidden behind props. The tank surface is facetted and has some buckles in it. To the far left of the vessel there is a smaller motor/ pump block covered by cooling gills and mounted on a socket; it is captured with a poor resolution.

Two experiment series were conducted with different *seed* region *partitions*. In the first series (figure 6-3), a partition was obtained from curvature signs; similar results were obtained using Koenderink's shape descriptor. Therefore, the cylindrical vessel part and its elliptical closure form separate regions. The STD algorithm models the cylindrical part first but realizes that the elliptical closure is its right SoR extension. By doing so STD suppresses any competing model hypothesis on the elliptical region, where FPT not using the SOR extension concept generates another cylinder hypothesis with wrong axis direction; their spatial conflict must then be dealt with by hypothesis evaluation.

The partition is 'over-segmented' as seen on the tank which contains also flat and even concave spots, and more so on the pump/ motor block. Only STD and FPT can create viable model hypotheses from so small cylindrical regions.

The second series shown in figure 6-4 uses classification by Mean Curvature sign solely discerning convex, concave, and flat areas. Curvature thresholds are set spatially adaptive guided by an initial partition into six levels of curvature magnitude. The entire tank now forms *one* region containing ridge and peak points; therefore, a straight-axis SoR with nonlinear radius function is appropriate. FPT detects a conical left part with linear radius function (though inaccurate) whereas Chain fails.

On the pump/motor unit the cylindrical motor and its socket dissolve in a single 'under-segmented' region. All algorithms are facing trouble processing this region, where STD using refined axis estimation scores best here. Accuracy of axis location and radius on the pump and motor block leave much to be desired for all algorithms.

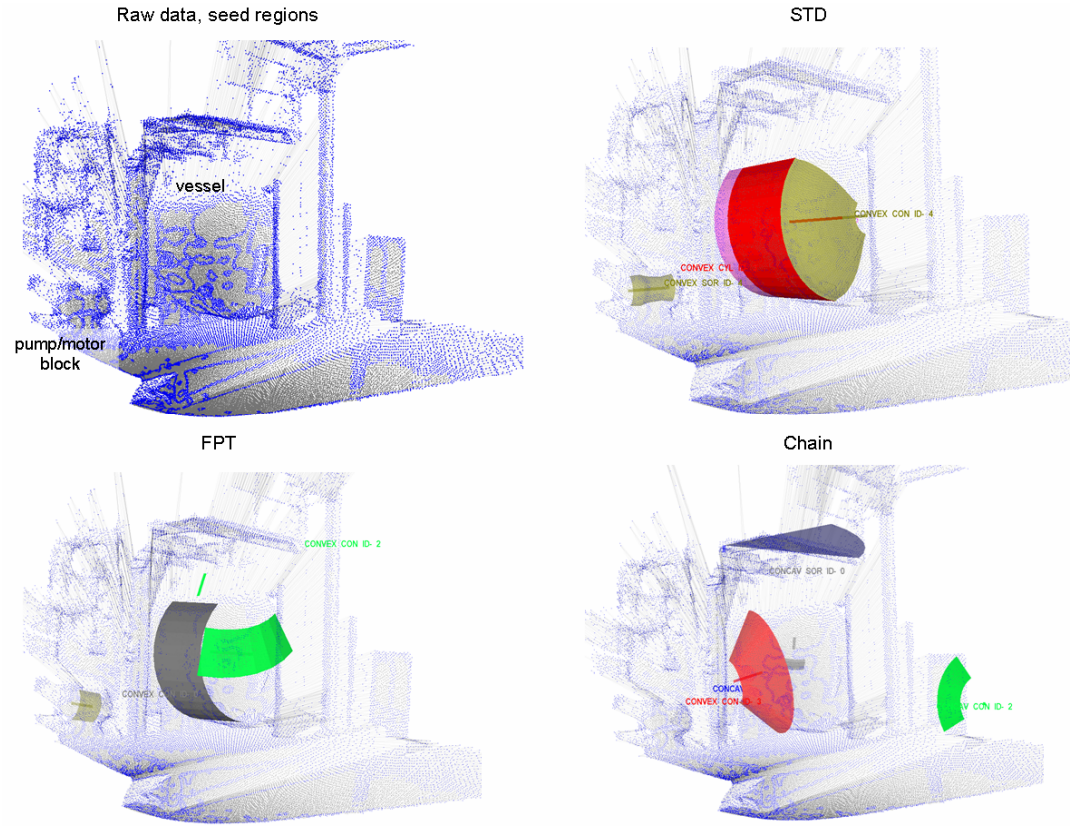

fig. 6-3 Detection and modelling results for scene type Kessel_2012 using 'over-segmented' seed regions

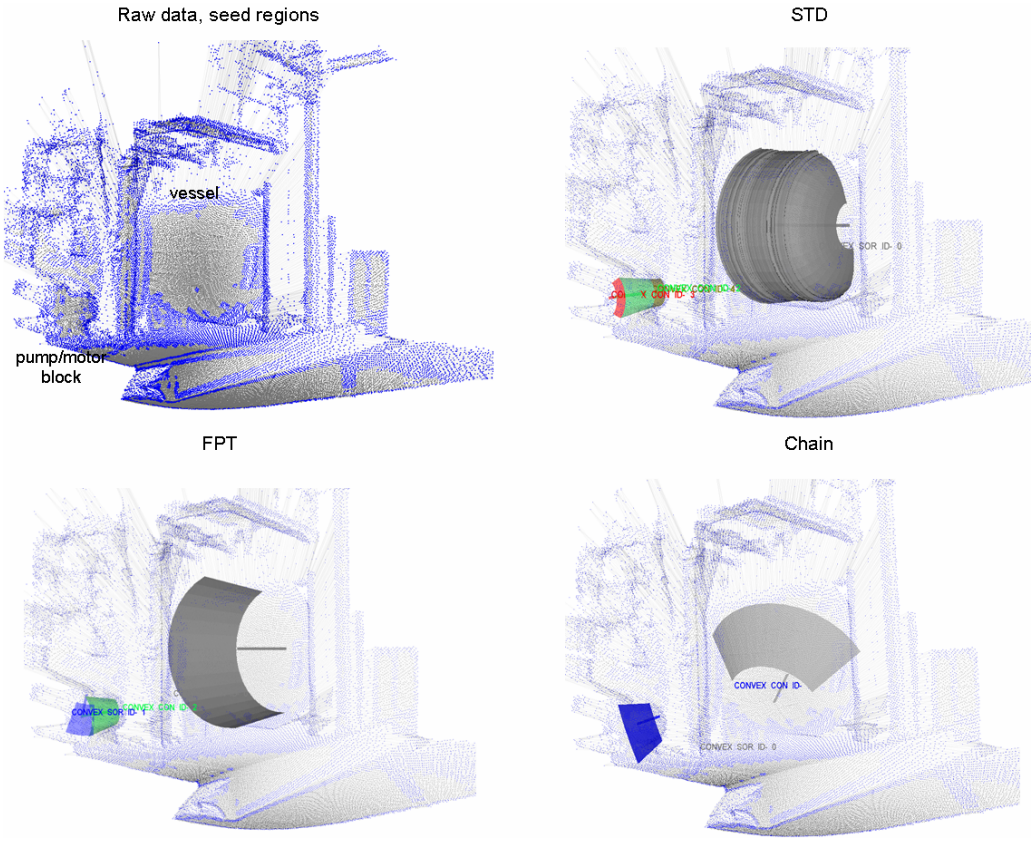

fig. 6-4 Results for scene type Kessel_2012 using 'under-segmented' seed regions

6.1.3 Flue gas washer system - III

A central section of the exhaust gas system is shown in the third type of scene. The large conical flue gas washer in the centre has been scanned from an upper level through a railing; part of it is seen as an obstacle closely in front of the sensor (marked in the raw data in figure 6-5 and suppressed in the remaining images). On the bottom, the flue gas washer has a vertical outlet going into the cuboid-shaped casing. Right of the washer extends a complex feed pipe system; its gross shape consists of a short vertical duct and a longer curved horizontal duct. Auxiliary objects are attached to the washer and to the supply pipes; the large pipes are also flanged. These 'decorations' are treated as noise as they are not sufficiently resolved from a distance of close to 8m. Downward from the supply duct runs another small and complex-shaped strand ending in a small flask-like vessel. The scene background is cluttered by further pipes and stanchions of various cross-sectional shapes. Part of the roof construction divided into tub-shaped sections is also visible.

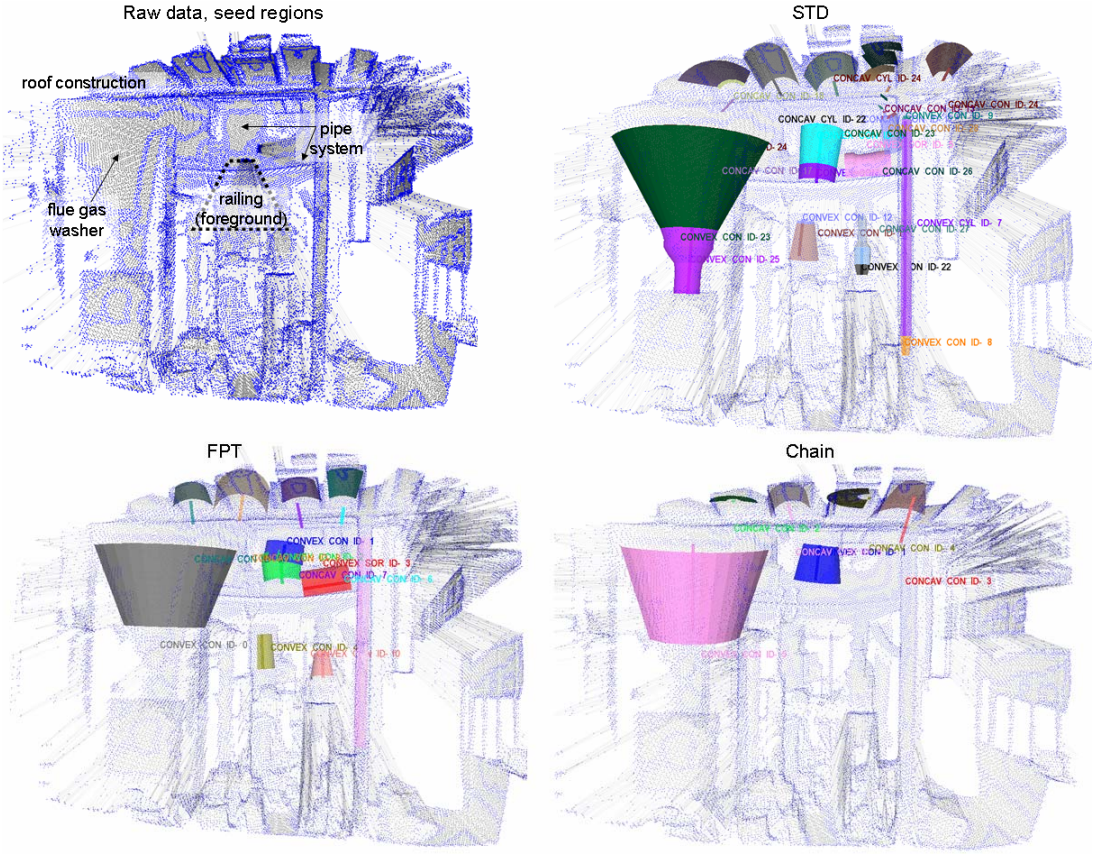

fig. 6-5 Detection and modelling results for scene type Rauchgas

The STD algorithm detects and models the few biggest parts, the washer, part of the feed system, and few auxiliary objects correctly. The washer cone axis and the opening angle (≈30°) can be considered accurate when the downward extension to the concentric outlet succeeds. This often happens with the STD algorithm, due to the good performance of the initial Plücker axis estimation, but rarely for FPT and Chain. EM optimization cannot iron out a poor initial estimate using only the geometric distance as a fitting error.

With the seed region partitions currently provided, even STD does not model the curved supply pipe system as a contiguous duct, but detects parts on it. On the other hand, STD finds plenty of - by human judgement - 'false positive' examples, such as stanchions modelled as convex cylinders, as well as many concave cylinders in the roof structure.

6.1.4 Rotary kiln - II

The scene in figure 6-6 shows the heart of thermal processing plant, a rotary kiln. Its left cylindrical section, ≈3000mm long and 2000mm in diameter seen from below at a fairly close range (2-3m), is separated from the right part by a gear rim. This is the only scene view entirely dominated by one large region containing about 12000 points. Below the rotary kiln we see clutter from metal coverings and small auxiliary parts, some of which have rotation symmetric surfaces, others have rounded planar or free-form surfaces. Viewing the low spatial resolution and using several pixels wide windows for extracting normal and curvature features, it is impossible to isolate such small objects and recognize them as planar but perceive the low-curvature kiln mantle as cylindrical.

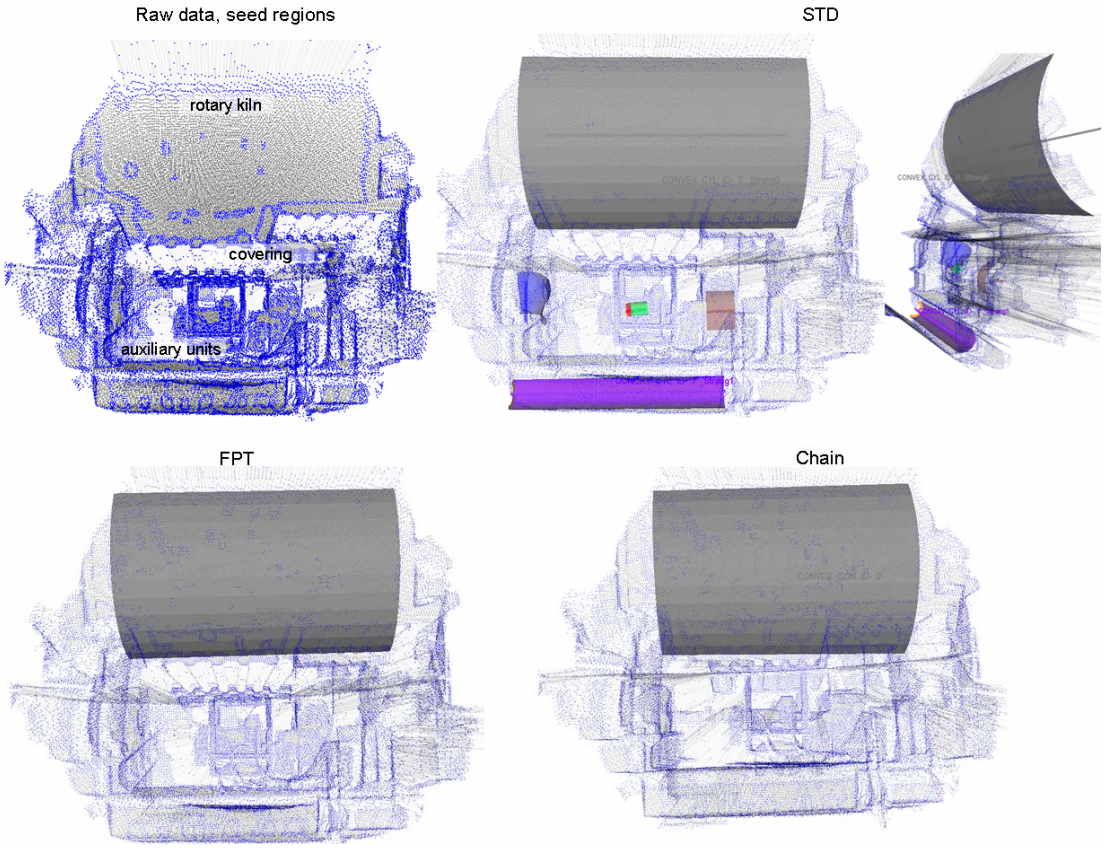

fig. 6-6 Detection and modelling results for scene type Drehrohr

All algorithms model the kiln surface correctly and repeatable, yet only STD achieves a satisfactory relative accuracy <1% in the axis, radius, and opening angle parameters. FPT underestimates the radius by ≈5%, and Chain estimates a cone angle of ≈2°.

The few results obtained for the auxiliary objects below the kiln are not considered useful.

The reader may notice the concave (hollow) 'pipe' on the bottom detected by the STD algorithm. Between the scanning device and the platform of the rotary kiln there is a gap (cavity); the points from this region are characterized as being of valley, and not flat type. Therefore, a concave SoR (hollow cylinder) is fitted that survives the hypothesis evaluation. Although an alternative model of three adjacent planes is successfully fitted at the evaluation stage, it remains weaker than the cylinder hypothesis.

6.1.5 Pressure reducer - IV

This scene in figure 6-7 is cluttered and contains shelves, props, stanchions, and wired strutting. In the bottom-left area is seen the rotation object of main interest, a pressure reducer or flash tank, connected to its feed system on the right via two horizontal pipes entering another vertical pipe. These feed pipes are roughly wrapped with thick sheet of isolating material and equipped with swivel valves; their surface has an irregular shape. The flash tank resembles a lying bottle and has a facetted surface with longitudinal grooves. Most of the rear end is occluded, and the visible front part is of partly ellipsoidal and partly hyperbolical shape. The

planar 'cap' of the tank facing the sensor has a rounded rectangular contour and extends further to the left, i.e. is not concentric.

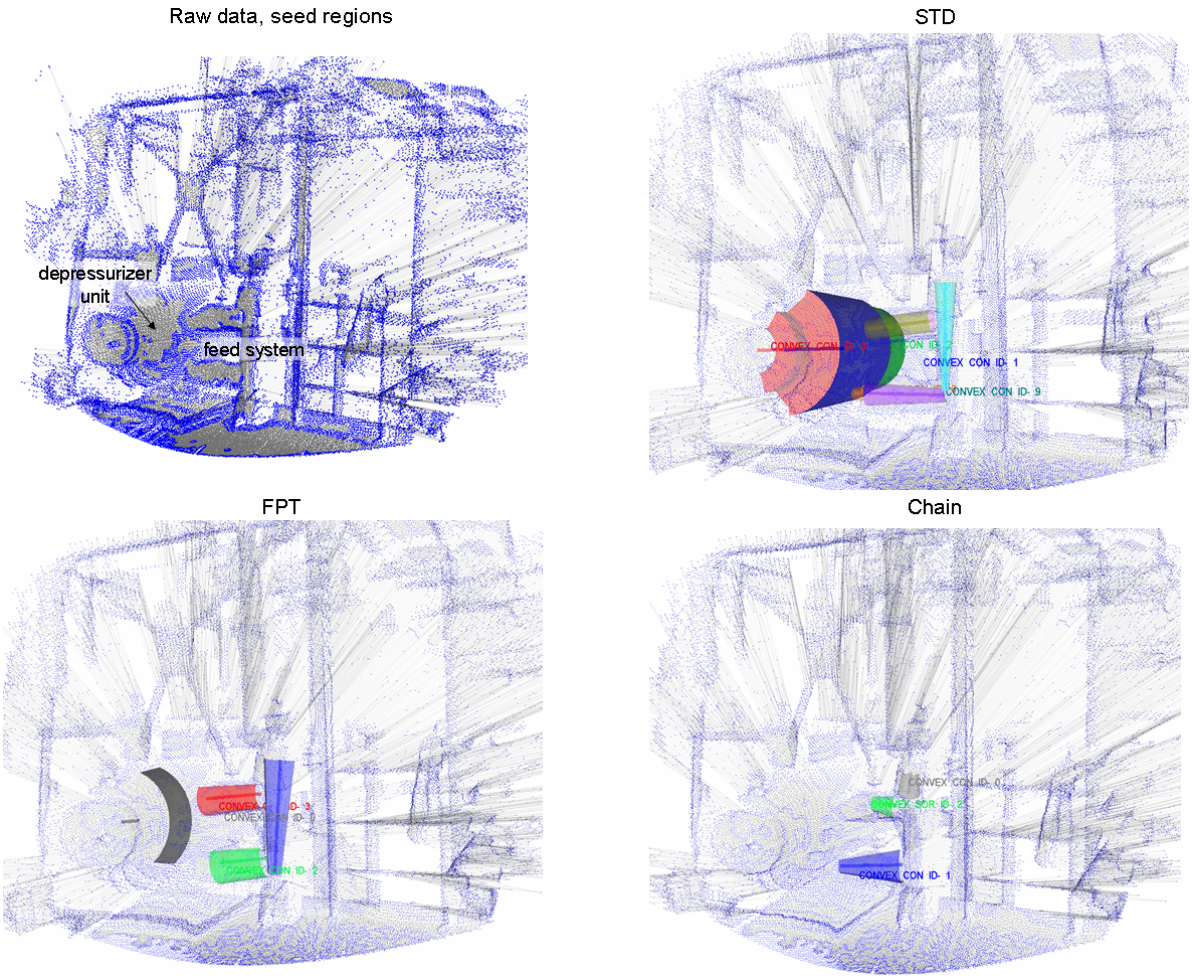

fig. 6-7 Detection and modelling results for scene type Entspanner_B2005

The flash tank is a rather unpleasant shape to process. Some satisfactory results were obtained using STD, but the main symmetry axis is not found repeatedly and accurately from the chosen view point. With this example the Plücker axis estimation (5.4.4) had the most problems, the reason lying in the ellipsoidal/hyperbolical shape with comparatively few and sparse points (in the background) indicating an elongated object. Therefore two close eigenvalues make it hard to disambiguate the true axis direction. In most runs, the axis estimate is skewed by 20-30° with respect to the true axis. EM algorithm fits a SoR surface to this axis and locks in on the local optimum with an acceptable fitting error. However, the performance measures of surface and region coverage remain poor, as seen from the broken surface appearance in figure 6-8. In some cases, those figures are low enough to give the Principal Curve algorithm a second chance to re-process the foot point cloud. Indeed, the alternative algorithm comes up with a different and better aligned axis (5-10°). EM then fits a cylinder and also a left/right SoR extension which fits well except near the mentioned cap surface.

To estimate the feed system, it turns out important to offer it as a single 'under-segmented' region. Figure 6-8 shows several decompositions into straight axis parts by section 5.4.5. It is to note that the part details, mostly the radius function approximations and the break points

between the pieces are inaccurate and sensitive to noise and random sampling. Yet, a satisfactory approximation of the gross shape and dimensions is achieved.

FPT fits a misaligned truncated cone to the visible tank surface, and the Chain algorithm fails; both algorithms are restricted to linear radius functions. On the feed system, FPT accomplished a coarse approximation by cylinder pieces, but not so the Chain algorithm because most cross-sectional chains are too short to generate hypotheses.

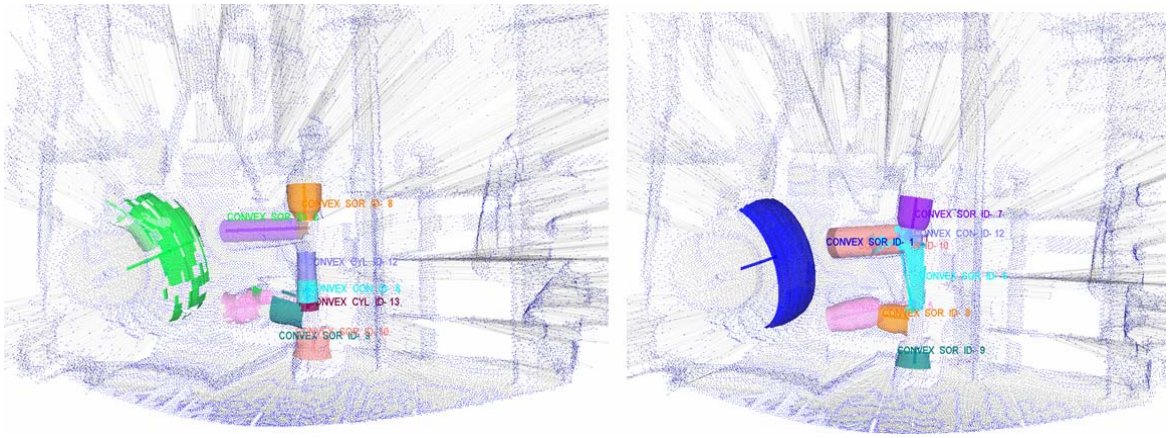

fig. 6-8 Different decompositions of the feed system in scene view Entspanner_B2005

6.1.6 Water pipe system B2005 - III

This rather 'airy' view (figure 6-9) shows a crowded assembly of water pipes, running horizontally and vertically next to the pressure reducer. Unlike all previous scene views, there are no planar floors or ceiling surfaces, and no seed region of a rotational surface has more than 600 points[11]. Thin pipes are resolved by 8 points in width or less. The laser scanner distance error becomes non-negligible compared to the radii. A steeply growing angular uncertainty (NPA angles, section 5.4.5) is therefore inevitable.

With STD, most pipes and also some beams are extracted as contiguous segments. The parameter estimates are reasonably accurate: all convex cylinders have an estimated opening angle below 2° and radii between 35mm and 60mm (pipe diameter 80-100mm?). Because of the high angular uncertainty, it is impossible to verify or falsify narrow cross sections as being circular. A prop with a square cross section is easily mixed up with a pipe, and a beam with U-shaped cross section may be interpreted as a hollow (concave) half pipe.

FPT achieves similar results, but loses some details, i.e. has more false-negatives. With the Chain algorithm, contiguous pipes are fragmented into segments, because of disconnected MIN and MAX chains.

[11] In the work [Rab06] at least 2000 points per pipe were required.

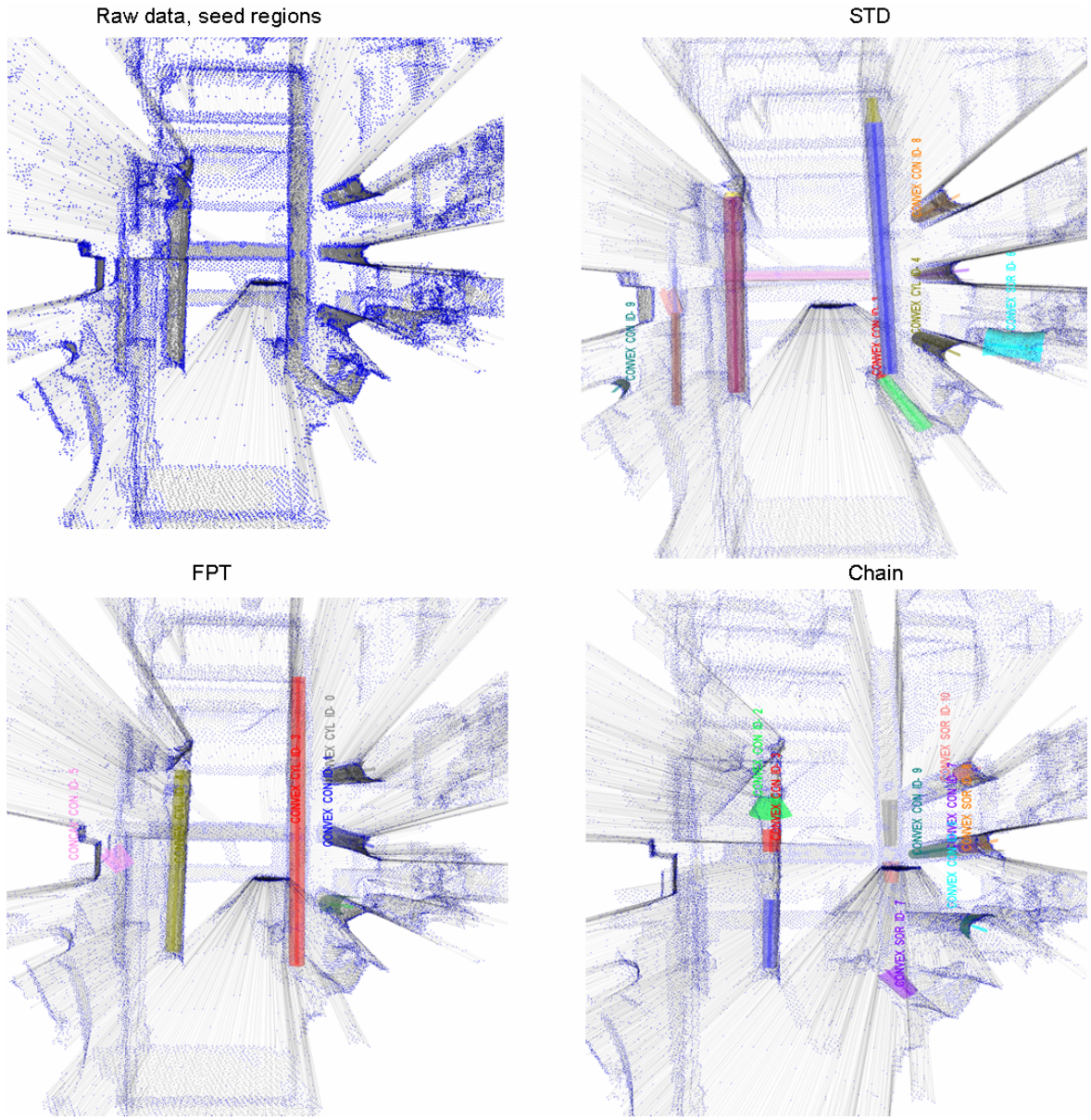

fig. 6-9 Detection and modelling results for scene type Wasserleit_B2005

6.1.7 Stirring unit R2010 - III

This is another cluttered scene view capturing 'bits and pieces' next to a plant assembly called agitator or stirring unit (Rührwerk). At least four cylindrical objects should be estimated: two barrels with vertical axis denoted barrel 1 (partly occluded, on the left in figure 6-10), barrel 2 (the largest one in the centre-background), and two vertical steel cylinders enclosed in a metal cage and partly obscured (cylinder nr. 3 - an ammonia dosing tank on top and cylinder nr. 4 on the bottom in figure 6-10). Estimating cylinder 4 is especially challenging since a relatively small portion of it is only visible.

These four objects are detected by all three algorithms. Reasonably accurate and consistently stable axis directions have only been obtained with the STD algorithm. As to the radii, the STD estimates sometimes differ by a factor of two; cylinder 4 is mostly estimated as the smaller one although steel cylinders 3 and 4 appear to be of equal size.

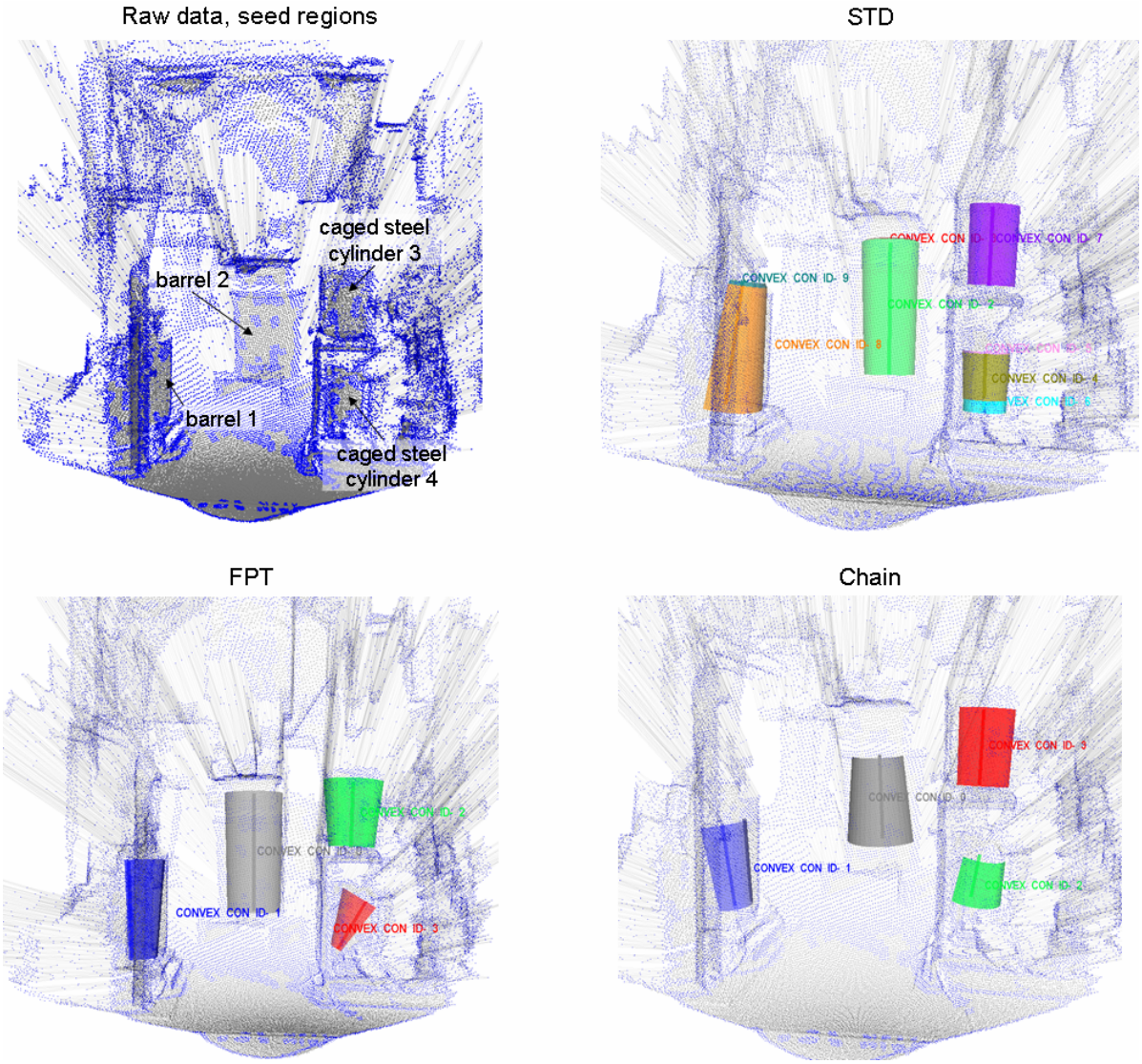

fig. 6-10 Detection and modelling results for scene type Rührwerk_R2010

Before testing on the THERESA images, confidence tests were performed with the Chain algorithm on range images of simple curved test objects kindly supplied by the Michigan State University in a public database. A few more tests were done on range images from a different laboratory captured with our own scanner. Some cylinder detection results are visualized in figures 6-11. In fact, most estimates are reasonably accurate, e.g. the axis lines estimated independently for different segments of the pipe junction isbigwy1 were verified as being coplanar, respectively, coaxial to a high degree of confidence (>0.96), and the pipe branching relation is also safely detected. Even the Chain algorithm in combination with EM is operational; its apparently weak performance, lacking robustness and accuracy on the THERESA images is therefore not probably due to an obvious flaw in the method, its implementation, or the raw data.

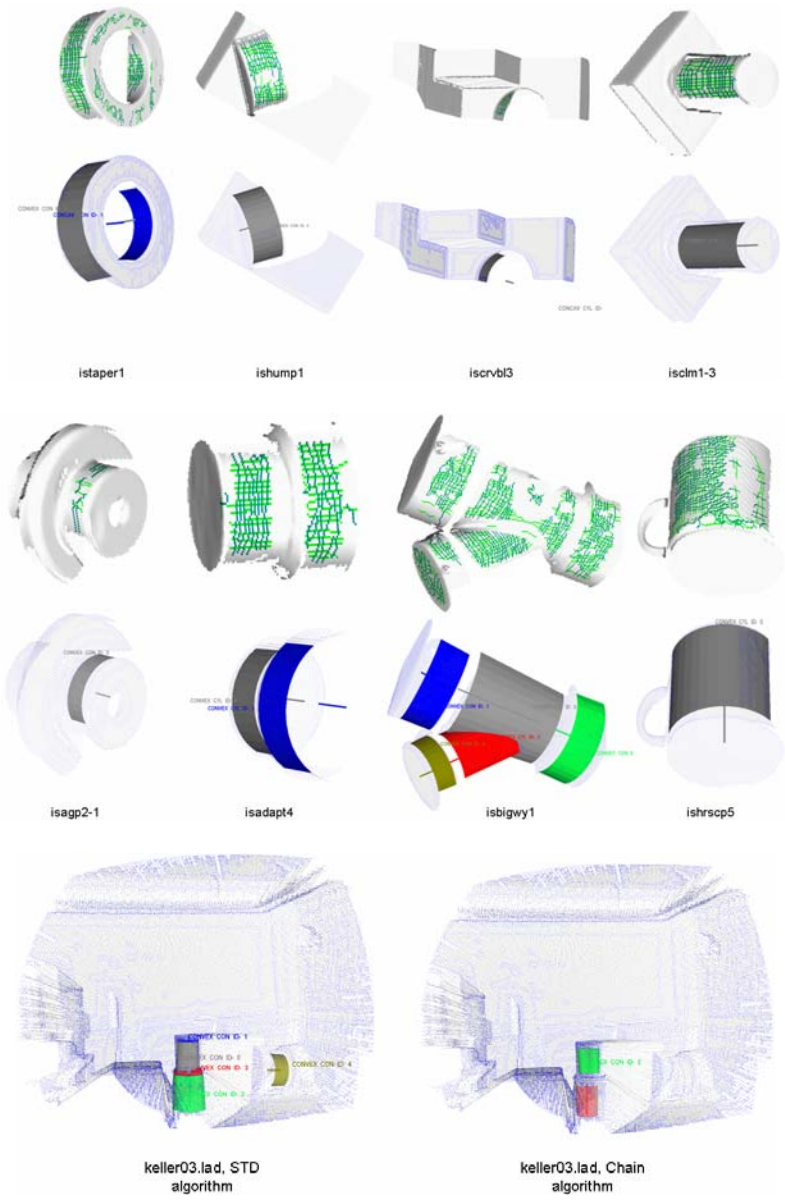

fig. 6-11 Test results of the Chain algorithm for simple test objects / scenes (see text)

6.1.8 Accuracy and error rates

Table 6-1 summarizes accuracy measurements of several object models; between 50 and 100 randomized runs were performed using the STD estimation on each object. The residual fitting error (column 3) is mainly governed by the law of large numbers; i.e. small residuals correspond to large support regions and vice versa; the size of the final support set is given in the rightmost column. Relatively large residuals are observed on a few large objects also such as the tank (Kessel_2012) which is not truly rotationally symmetric but bruised and facetted, and on the flue gas washer (Rauchgas) which is not entirely conical but has an auxiliary object attached to it on one side which is treated as noise.

Scene view	Object	Residual rms [mm]	σ_D [mm]	σ_φ [°]	$\bar{r}$ [mm]	σ_r [mm]	$\bar{\delta}$ [°]	σ_δ [°]	r_{GT} [mm]	δ_{GT} [°]	nr. of pts
Baugruppe_6201	Bypass Upper Cyl.	17,43	26,0	0,7	317	17,9	-0,0	0,2	380?	0	973
Baugruppe_6201	Bypass Kink Hor.	8,99	55,7	2,2	486	10,7	12,2	12,5	500?	0	435
Baugruppe_6201	Bypass Kink Ver.	14,76	78,3	6,8	398	46,9	-0,8	7,1	420?	0	391
Baugruppe_6201	ID Fan Upper Cyl.	11,32	48,7	0,5	390	30,0	4,9	0,4	450?	0	1643
Baugruppe_6201	ID Fan Outlet	17,12	2,9	0,6	83	1,1	-0,2	0,2	100?	0	511
Baugruppe_6201	Rear Duct	9,42	175,9	4,6	697	104,6	-5,8	6,4	550?	0	373
Kessel_2012	Tank Cyl.	12,99	95,5	1,2	1026	12,2	-2,1	3,4	1050?	0	3047
Kessel_2012	Tank SoR	11,83	79,4	13,1	928	28,1	-	-	-	0	3710
Rauchgas_1	Flue gas washer	15,48	30,4	1,7	896	29,2	30,7	4,2	-	30?	1651
Rauchgas_1	Feed vertical cyl.	17,59	89,2	4,4	325	92,6	3,6	4,7	400?	0	454
Drehrohr	Rotary kiln	5,48	10,6	0,8	1003	9,9	0,7	0,6	1000?	0	13255
Wasserleit_B2005	3 parallel pipes	8,41	7,02	4,9	56	3,9	1,5	0,4	120?	0	1210
Rührwerk_R2010	Barrel 1	18,41	12,89	1,3	149	10,2	3,7	1,1	250?	0	1283
Rührwerk_R2010	Barrel 2	26,14	20,62	0,8	200	15,7	-1,5	1,2	250?	0	847
Rührwerk_R2010	Ammonia Cyl.	12,30	47,52	5,9	149	13,8	-3,4	2,4	200?	0	774
Rührwerk_R2010	Lower Steel Cyl.	9,75	31,94	2,7	133	33,1	5,0	10,8	200?	0	540

Table 6-1: Accuracy measurements on selected THERESA objects using the STD algorithm. Legend: Residual denotes root mean square of the fitting error, σ_D, σ_φ: standard deviations of the axis distance D [mm] from the origin respectively azimuth angle φ [°] describing the axis direction. $\bar{r}$, σ_r: mean resp. std dev of the estimated average object radius in [mm]. $\bar{\delta}$, σ_δ: mean resp. std dev of the estimated opening angle in [°]. r_{GT}, δ_{GT}: ground truth radius resp. opening angle of object.

The residual says little about the *accuracy* of parameter estimation, be that measured as the variance or standard deviation of parameter estimates (σ_D, σ_φ, σ_r, σ_δ) or as the bias with respect to ground truth object dimensions which we do not know precisely. Accuracy depends on the conditioning of the estimation problem, in particular on the radial angle subtended by a patch from which a SoR radius is estimated[12]. For example, accuracy of the rear duct in Baugruppe_6201 is low because of its poor visibility, despite a rather small residual error. Accuracy also depends on the complexity of object geometry and the quality of the seed region provided. Table 6-1 also indicates that, with few exceptions, the radii of SoR objects are consistently underestimated from single views by 10-20%.

A further experiment series was conducted to see how many hypotheses are processed in different algorithm stages, to analyze the efficacy of hypothesis verification and to estimate the number of false-positive and false-negative models. Some figures obtained for the STD algorithm are reported in table 6-2; the counting variables are explained in table 6-3.

[12] Conditioning is illustrated by the problem of 2D fitting a *circle* to point samples lying on a *small-angle arc* [ZeC04]. It refers to the *amplification* of data uncertainty as a function of a hidden parameter describing the input data, here the subtended arc angle. In the 3D case, estimating the SoR axis position and radius function from a surface patch is like estimating many concentric 2D circles (centers, radii), assuming that the axis direction is known. If the input patch has ambiguous principal directions because of close or multiple eigenvalues, yet another dimension of ill-conditioning comes into play. How to compactly characterize a surface patch in order to predict the estimation accuracy is an open issue.

NR	NH_G	NH_{Opt}	NH_{FPT}	NM_{SoR}	NM_V	NM_{VR}	NM_{fp}	NM_{fn}	Range view
47	43	14	0	19	10	10	3	1	Gruppe_6201_Vw0
35	33	15	0	24	10	10	1	3	Gruppe_6201_Vw17
34	29	14	0	23	14	13	4	2	Gruppe_6201_Vw19
51	43	16	0	24	17	16	11	1	Rauchgas_Vw0
53	46	15	0	23	13	13	6	1	Rauchgas_Vw1
46	44	23	0	38	24	20	15	2	Rauchgas_Vw25
48	38	21	0	33	18	17	11	1	Rauchgas_Vw26
50	46	22	0	36	27	24	18	2	Rauchgas_Vw27
16	20	13	1	18	6	6	1	0	Kessel2012_Vw0
18	18	13	5	23	13	9	3	1	Kessel2012_Vw2
40	34	20	0	26	18	18	13	2	Drehrohr_slow0
34	28	19	0	23	15	15	11	2	Drehrohr_slow1
49	44	28	1	38	18	17	10	0	Ruehrwerk_R2010_Vw34
45	38	15	0	19	11	10	5	1	Ruehrwerk_R2010_Vw35
38	34	25	0	30	28	28	12	3	Wasserleit_B2005_Vw3
31	28	19	0	23	21	21	10	3	Wasserleit_B2005_Vw4
35	32	21	0	28	26	26	17	3	Wasserleit_B2005_Vw5

Table 6-2: Evolution of the number of hypotheses

Variable	Meaning
NR	Number of component regions selected for SoR hypothesis generation
NH_G	Number of hypotheses generated using direct methods
NH_{Opt}	Number of hypotheses completing the EM optimization
NH_{FPT}	Number of hypotheses generated using alternative FPT method
NM_{SoR}	Number of SoR models (linear axis segments) after optimization
NM_V	Number of SoR models after hypothesis verification
NM_{VR}	Number of SoR models after relation generation
NM_{fp}	Number of false ***positive*** SoR models by visual inspection
NM_{fn}	Number of false ***negative*** SoR models by visual inspection

Table 6-3: Meaning of the variables in table 6-2

In the first three columns the hypotheses spawned by disjoint seed regions are counted. The number NR of regions initially selected therefore exceeds or equals the number NH_G of hypotheses generated which itself bounds the number NH_{Opt} completing optimization. Alternative hypotheses (NH_{FPT}) are generated if a high unused potential remains after optimization. According to column 4 in table 6-2, few of those are generated and only in some scenes. NM_{SoR} counts the number of SoR models (parameter vectors) with linear axis. Each hypothesis completing the optimization generates at least one such model, therefore $NM_{SoR} \geq NH_{Opt}$. Columns 5 to 7 illustrate the hypothesis pruning by the first verification step (section 5.6) and the final relation estimation (section 5.7); clearly $NM_{SoR} \geq NM_V \geq NM_{VR}$. Some illustrative examples of hypothesis pruning are shown in figure 6-12. The numbers of false-positives (phantom models) and false-negatives (missed SoR objects) were determined by visual inspection and using our knowledge of the plant. Therefore NM_{VR} + NM_{fp} - NM_{fn} should equal the 'true' number of SoR models by human expectation, regardless of whether sufficient scan data were actually available to detect them. For many smaller objects this is not the case.

As becomes obvious from table 6-2, many false-positive models remain at the end, most of which receive low final quality and weight. A low acceptance threshold was deliberately set not to suppress too many. A more realistic evaluation count would consider only those models as false-positives whose quality is higher than at least one true-positive model in the output.

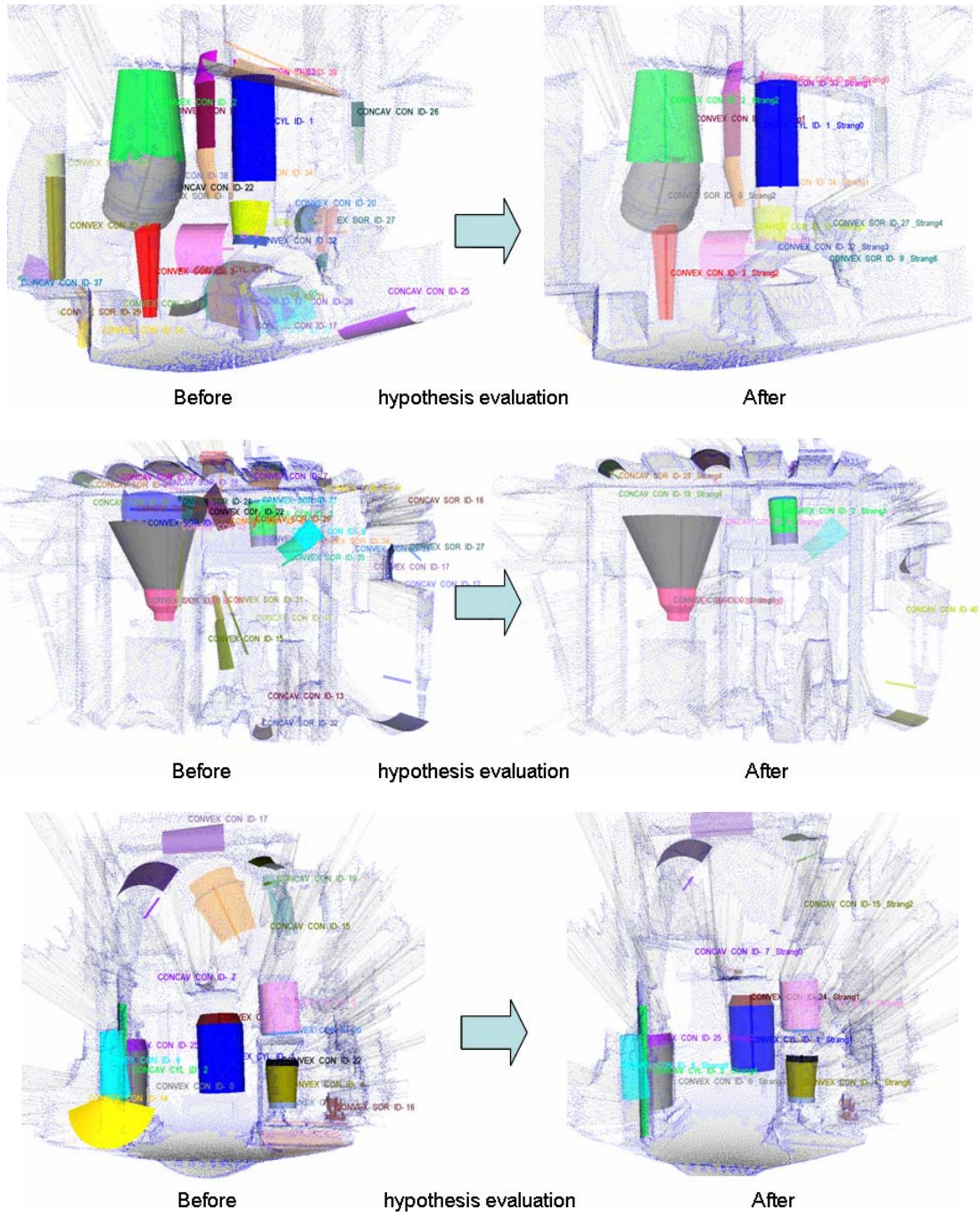

fig. 6-12 Hypothesis decimation by evaluation and relation building, illustrated at the range views Baugruppe_6201_vw0 (top), Rauchgas_vw0 (middle) and Rührwerk_vw18 (bottom).

6.2 Stability assessment

To what extent will the algorithm results be affected by minor variations of the input data assuming that they still come from the same object geometry? This is the important *stability* (or *repeatability*) issue to be investigated next. The input distribution need not be mathematically defined as for *sensitivity* analysis; it may reflect subjective judgment on the kind of variations to be tolerated. Therefore, the resulting output variations have no meaning as absolute fig-

ures but allow quantitatively comparing different algorithms subjected to the *same* input variations. Stability assesses the *inner* variation of algorithm output; it pays no regard to *external* guidelines of correctness ('ground truth'), and should not be confused with *accuracy* measuring deviations from *expected* output, or with *qualitative performance* measures such as false-positive or false-negative rates or confusion matrices for learning classifiers. A stable algorithm may give consistently wrong results. The converse, an unstable algorithm giving correct results, is not true, at least not in a statistical sense.

6.2.1 Experimental design

At first, the input variations must be specified. All experiments described here use real range images captured from a thermal processing plant, not simulated data. The input data of the hypothesis generation and optimization algorithms are connected components (regions) of pre-processed point features. These regions, being samples from some common geometry, were modified and *additionally* disturbed in the following ways:

1. Random point sampling: from the N_0 points in a region only subsets of $N<N_0$ points are drawn at random and offered to the algorithm. Point indices are repeatedly drawn with *uniform* probability from an index interval $[1..n]$ ($N<n\leq N_0$) and then removed, which means *sampling without replacement*[13]. The point *addresses* referring to the range image and allowing access to the calculated point features remain part of the subset. Thus, the *organization* (scanning topology, neighborhood) of points is not lost and may be used by candidate algorithms. But no equidistant sampling property will hold.

2. Adding Gaussian noise: Independent, normally distributed errors $N(0, \sigma)$ with zero mean and uniform variance σ^2 are added to all point coordinates. Adding Gaussian noise mainly simulates different levels of scanner accuracy. Unlike range sensor models used in probabilistic robotics [TBF06], no exponentially distributed term for the likelihood of false readings in free space has been included here.

3. Adding motion noise: Platform motion uncertainty becomes non-negligible as the range image capture times are relatively large. Using current laser scanner technology, the range image 'frame rate' is in the order of one image per second, one to two orders of magnitude less than with conventional visual SLAM.
   It is not assumed that the carrier platform (robot, cart, or human) stands still during image capture, although the test images in this study are more or less still images. What we however do assume is *constant* and *known velocity* of translation and rotation during image capture, available from the inter-frame ($i\rightarrow i+1$) estimates of rigid observer motion $\mathbf{T}_i(\mathbf{x}) = \mathbf{R}_i \cdot \mathbf{x} + \mathbf{t}_i$, and from the time interval ΔT_i between consecutive image captures. With $\mathbf{R}_i := Rot(\mathbf{a}, \alpha)$ and assuming piecewise constant velocity of translation $v \approx \|\mathbf{t}_i\| / \Delta T_i \; [m/s]$ and rotation $\omega \approx \alpha/\Delta T_i \; [rad/s]$, the observer pose for an object captured t seconds after the image start at pose $\mathbf{o}_i$ can be corrected ($Rot(\mathbf{a}, \omega \cdot t) \cdot \mathbf{o}_i + v \cdot t \cdot \mathbf{t} / \|\mathbf{t}_i\|$) and the motion thereby compensated.

[13] In this way, any point appears at most once in the sample set. Resampling methods are commonly called *bootstrapping* in statistics which means, however, sampling *with* replacement.

Motion noise simulates *violations* of these assumptions, i.e. unknown platform accelerations and inaccurate velocity estimates. The motion noise specification includes Gaussian variations of

a. the translation velocity $\varepsilon_v \propto N(v_0, \sigma_v)$,
b. the angular velocity $\varepsilon_\omega \propto N(\omega_0, \sigma_\omega)$, and
c. the translation *direction* and rotation *axis* with regard to set values $\mathbf{t}_0$, $\mathbf{a}_0$.

The case $v_0=0$, $\omega_0=0$ models a 'correct' motion estimate with unknown stochastic accelerations due to skidding, slipping, bumping, or smooth transitions. A nonzero mean simulates biased motion estimates causing systematic image distortions (figure 6-13). Integrating velocity over time steps accumulates positional errors, i.e. image distortions keep growing from the first to the last scan line.

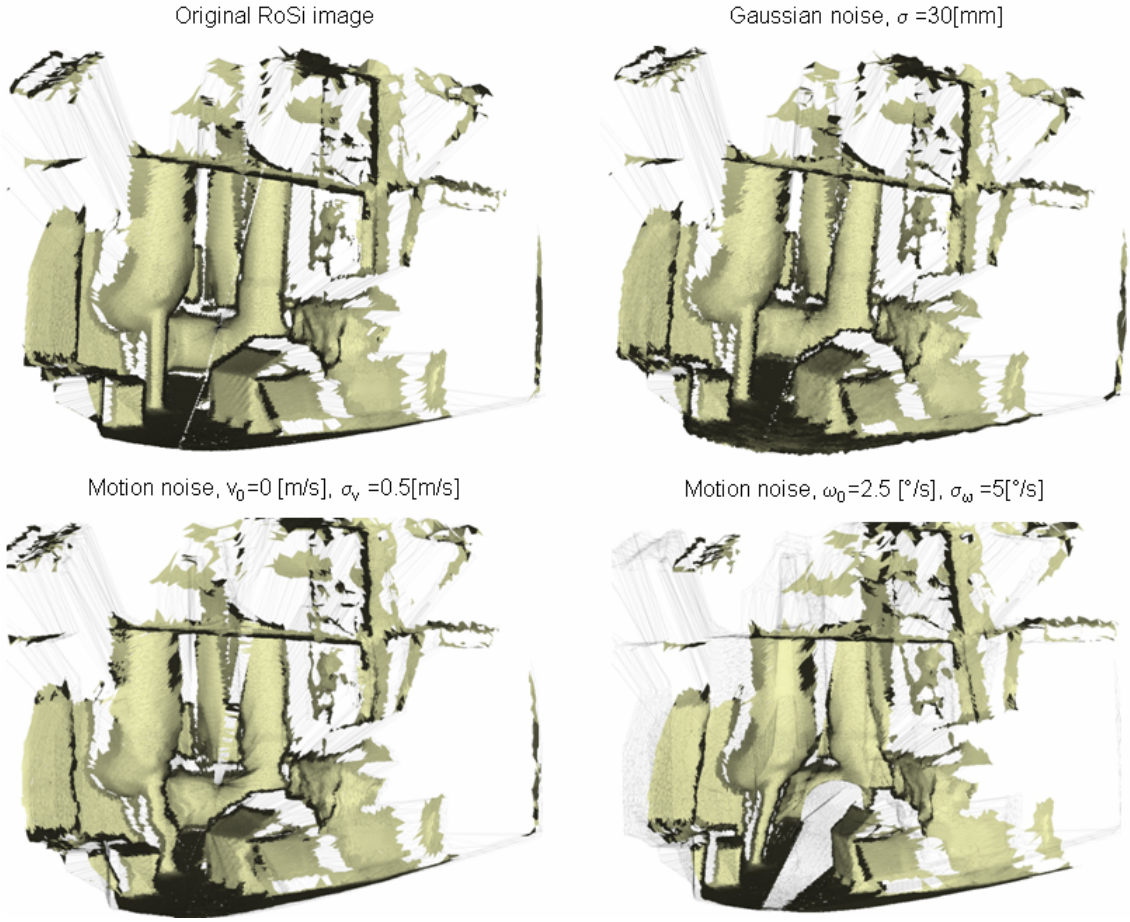

fig. 6-13 Adding different types of noise to a range view (Baugruppe_6201, top left): Gaussian noise (top right), Motion noise - unbiased translation within the floor plane (bottom left), and biased rotation about the floor normal (bottom right).

From a systems point of view, the random sampling 1) is simpler than the noise generation 2) and 3) as it solely affects the SoR estimation algorithms. Adding noise affects the entire preprocessing pipeline and in particular component labeling which, on its part, is fed into the estimation algorithms. How does hypothesis generation itself cope with the noise, and what is due to prior stages 'spoiling' the input, viewing figure 6-14? To better isolate estimation performance, the same *reference* component partition is imposed back after calculating the noisy point features.

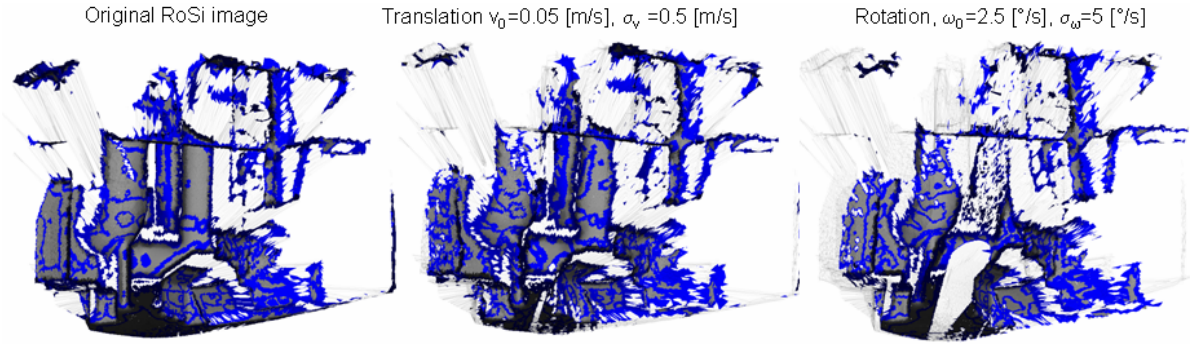

fig. 6-14 Motion noise affects the region partition and destroys the region structure.

Next, the assessment of algorithm output is specified as to how compare and quantify variations. As the output contains complex object features, this is not entirely straightforward. There are structural differences to be compared: the number of primitives expected is not known in advance, the number returned by an algorithm varies, including zero (failure). So do the types of primitives, i.e. cylinders, cones, or general SoR with polygon approximated radius functions. Also the estimated primitives are not equally important as to object size or number of points supporting them; this should be considered as well in the assessment.

From the viewpoint of stability assessment, the estimation algorithm maps input distributions to output distributions: $\boldsymbol{S_I \rightarrow S_O}$ [TCB08], i.e. variations in point features and regions "while sampling from essentially the same geometry" to variations of primitive parameters estimated. Stability is however hard to gauge from properties of *general* distributions. For easy and efficient experimentation, one should use basic distribution parameters such as the mean, covariance, or standard deviation. But the immediate evidence of covariance is lost as the parameter distribution is in general multi-modal and its shape depends on the scene geometry and the input data. For example, if a component captures a pipe *junction* or a *bent* pipe, each part forms a different peak in the distribution of parameter values. To use covariance as an indicator of stability, different distribution modes should be analyzed separately. Therefore, the random algorithm output needs to be decomposed first and its parts be assigned to their proper modes, based on similarity considerations.

This association, part of the evaluation system and not the test object, should be kept simple as it might adversely affect (obscure) the assessment results, otherwise. It can be simplified by exploiting the *self-assessment* of the candidate algorithms: each method *orders* its results (SoR parameters) by decreasing *weight*. To compare compound output, we let the *ordering or ranking number* of the parts *determine* its association: the highest weighted parameter vector contributes to the first mode, the one with second-highest weight to the second mode and so on. Thereby, two algorithms with similar numerical stability but unequally reliable self-assessment according to weight values will get different stability ratings. As the output ordering becomes unstable, the parts will more often get mixed up. The weight factors also serve as importance weights. To calculate the variance of a scalar parameter, say, the axis orientation angle α, its according variance $V_{\alpha,m}$ is found for each part m and the weighted sum $\sigma_\alpha^2 = V_\alpha = \sum_m V_{\alpha,m} w_m \left(\sum_m w_m = 1 \right)$ is formed.

The output representation used for assessment should present only *essential* parameters characterizing an object *uniquely* and avoiding random but meaningless variations. As a basis, we use the *Marshall* parameter vector K_M = (ρ, φ, θ, α, R, δ) for cones (sub-section 5.5.2.1) which should be appended by more (R_j, δ_j) pairs to account for piecewise linear SoR radius functions. A preliminary simplification has been made in order to deal with parameter vectors of constant dimension: the radius function is substituted by a regression line giving only two (intercept $\overline{R}$, slope $\overline{\delta}$) parameters. As the spatial extent is also essential to stability assessment, the minimum and maximum projection parameters s_{min}, s_{max} are added.

Summarizing, the random output of candidate algorithms for stability assessment has the form

$$\left(\left[\rho_1,\varphi_1,\theta_1,\alpha_1,\overline{R}_1,\overline{\delta}_1,s_{min,1},s_{max,1}\right]^T, w_1\right),\ldots,\left(\left[\rho_k,\varphi_k,\theta_k,\alpha_k,\overline{R}_k,\overline{\delta}_k,s_{min,k},s_{max,k}\right]^T, w_k\right) \quad (w_1 \geq w_2 \geq \ldots \geq w_k) \tag{6.1}$$

Where the number k of parts varies and the weights w_j indicate the sample weights of part j, $1 \leq j \leq k$. If an algorithm reports failure it returns no result sample. The number of result samples therefore depends on the algorithm and its rate of false-negatives and false-positives.

Standard deviations may now be obtained as square roots of the positive eigenvalues (variances) of 8x8 covariance matrices. The eigenvalues cannot be directly attributed to the scalar parameters, only their *sum* equals the sum of matrix eigenvalues, its trace. It is preferred to find a scalar variance for each parameter. To avoid assessing angular spread directly and dealing with circular angle distributions, we replace the three polar angles φ, θ, α of the canonical representation by two directions for the closest point $\mathbf{p}$ and the axis direction $\mathbf{a}$, respectively. The direction spread is measured by (square roots of) the two smaller eigenvalues of the 3x3 PCA matrix giving four values, e_{p1}, e_{p2} for $\mathbf{p}$ and e_{a1}, e_{a2} for $\mathbf{a}$. The resulting vector of deviations is Σ = (σ_ρ, e_{p1}, e_{p2}, e_{a1}, e_{a2}, σ_R, σ_δ, $\sigma_{s,min}$, $\sigma_{s,Len}$)^T.

6.2.2 Stability results

The stability experiments performed are characterized by three parameters

S1 Candidate algorithms: so far, the standard algorithm proposed in section 5.8 (STD), the chain algorithm (CHAIN), and a foot point transformation with adaptive mask size (FPT) have been compared.

S2 Object geometry: the component regions chosen as input, ranging in difficulty from well segmented cylindrical shapes to complex SoR with curved or multi-linear axis, including badly segmented regions of mixed types.

S3 Input variations according to the three types of noise described above and the level of noise applied.

All three candidate algorithms (S1) were 100 times on 13 different scene/region combinations (S2) and three different noise types (S3), yielding 3900 results per algorithm. To estimate the overall parameter standard deviation of each algorithm, the results from the 39 different experiments, i.e. scene and noise types, were averaged without weighting.

In order to compare the algorithm output in a single diagram covering parameters of different units and magnitudes, the deviations for each parameter were normalized by the group mean within the group of candidate algorithms. These results are shown in the figures 6-15 (deviations covering all noise types), 6-16 (deviations for random sampling), 6-17 (deviations for Gaussian noise), and 6-18 (deviations for motion noise).

At this lumped level of representation we may conclude

- On the average, the CHAIN algorithm shows the highest deviations, i.e. it scores lowest (worst) in stability.
- Considering only random point sampling as input modification, the STD algorithm shows the smallest output variations, i.e. it can be regarded as the most stable algorithm.
- Adding more Gaussian or motion noise, the advantage of STD begins to fade, and the foot point algorithm (FPT) draws almost level with STD; both perform significantly better than CHAIN.

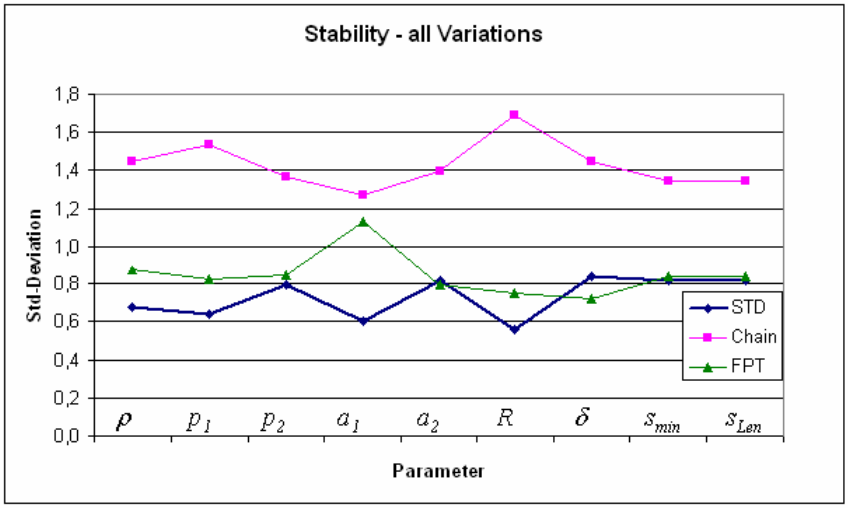

fig. 6-15 Relative stability of the algorithms STD, Chain, FPT under all input variations

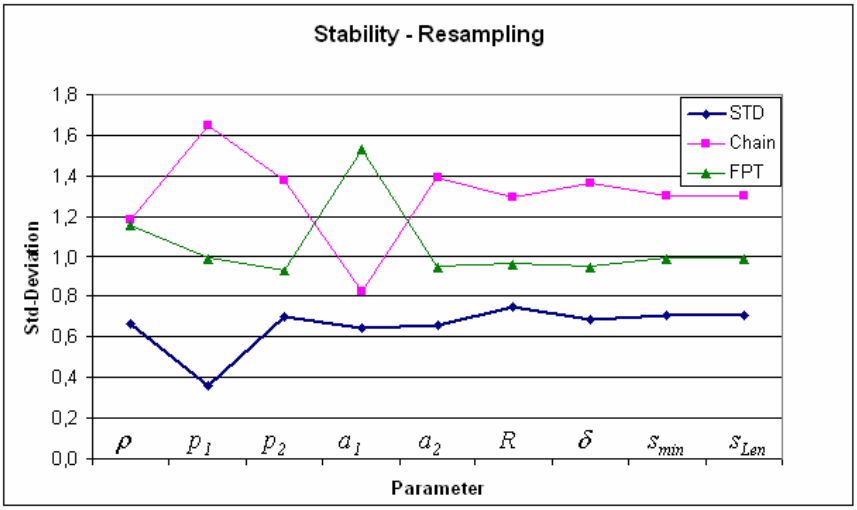

fig. 6-16 Relative stability of the algorithms STD, Chain, FPT under random input sampling

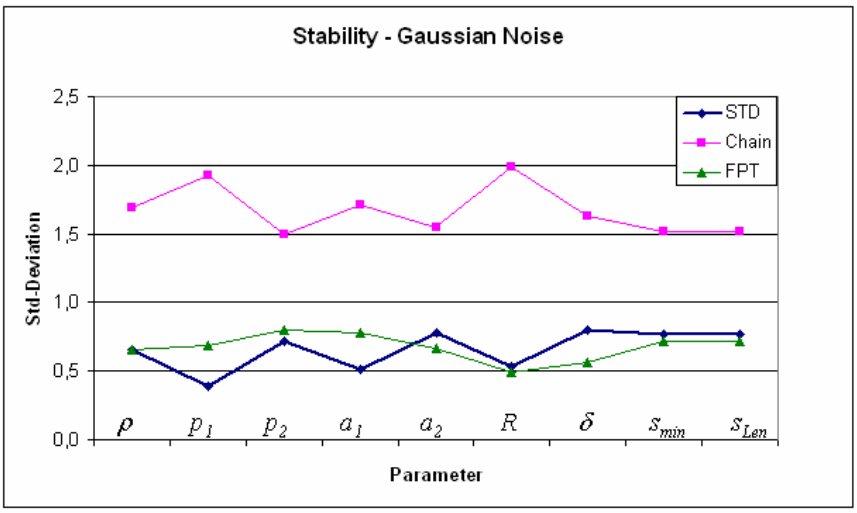

fig. 6-17 Relative stability of the algorithms STD, Chain, FPT under Gaussian noise

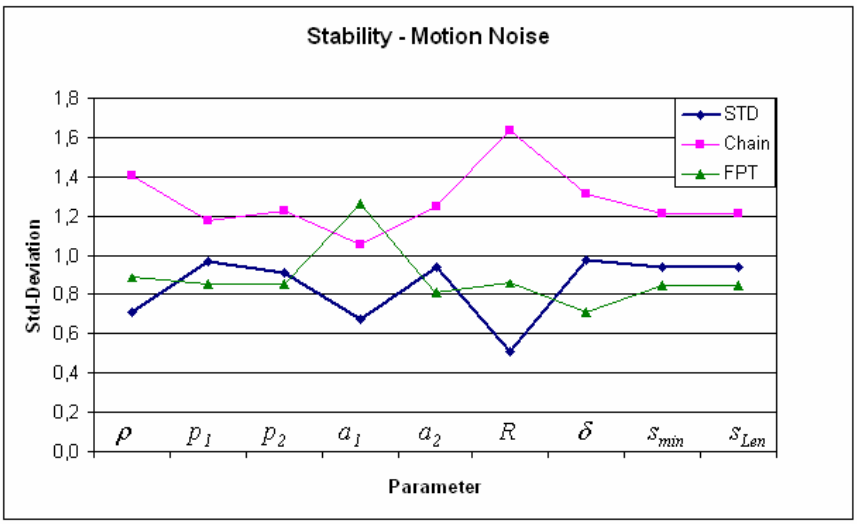

fig. 6-18 Relative stability of the algorithms STD, Chain, FPT under motion noise

To get an impression of the magnitudes for each parameter, absolute values of the standard deviations are listed in the table 6.4. As mentioned before, those are simply averaged from scene views containing mixed object shapes and sizes and different scanning densities, regardless of how often these cases occur or how important they are. Therefore each figure by itself has little significance as an 'accuracy' indicator. For example, the radius deviation σ_R = 73,78 mm listed in row 1 of table 6-4 averages extreme cases, including the densely sampled rotary kiln with $\sigma_R \approx$7,2mm (huge cylindrical region with ≈12000 points and an estimated radius of ≈1050mm) and also poorly resolved pipes with bent axes (regions containing less than 300 points) where deviations σ_R > 200 mm are seen.

Comparing between different cone and SoR parameters, the largest deviations are found in the projection distance s_{min} measured from the axis point closest to the origin as a reference.

The table also shows that motion noise is tolerated much less than Gaussian noise, and that no algorithm has a stable performance on scene views distorted by motion noise. It would be

a worthwhile task further refining these statistics according to different scene types, i.e. object geometries, region shapes and visibility properties.

Random Sampling	σ_ρ [mm]	e_{p1}	e_{p2}	e_{a1}[deg]	e_{a2}[deg]	σ_R [mm]	σ_δ [deg]	$\sigma_{s,min}$ [mm]	$\sigma_{s,Len}$ [mm]
STD	106,68	0,023	0,107	5,90	5,37	73,78	5,45	316,37	95,63
Chain	189,48	0,105	0,211	7,54	11,31	127,95	10,82	582,89	112,75
FPT	185,23	0,064	0,142	13,93	7,69	95,10	7,49	443,73	165,21
Gaussian									
STD	121,88	0,036	0,144	7,37	7,71	72,95	8,03	413,39	109,49
Chain	314,64	0,178	0,302	24,47	15,36	274,68	16,39	819,98	155,67
FPT	121,21	0,063	0,160	11,11	6,60	67,54	5,66	387,86	135,13
Motion									
STD	212,47	0,124	0,234	9,73	11,88	98,75	11,84	647,75	179,17
Chain	420,91	0,150	0,314	15,26	15,75	317,93	15,94	832,59	171,70
FPT	265,99	0,108	0,218	18,29	10,11	167,21	8,55	577,86	227,42

Table 6-4: Standard deviations of SoR parameters obtained with algorithms STD, Chain, and FPT.

So far, the method of stability assessment has been applied to compare three different algorithms for hypothesis generation. We may, of course, use it also to compare other algorithm parts, e.g. the method for parameter optimization (LM or ICP by section 5.5.2) used inside the STD algorithm. The result is illustrated in the following figure 6-19.

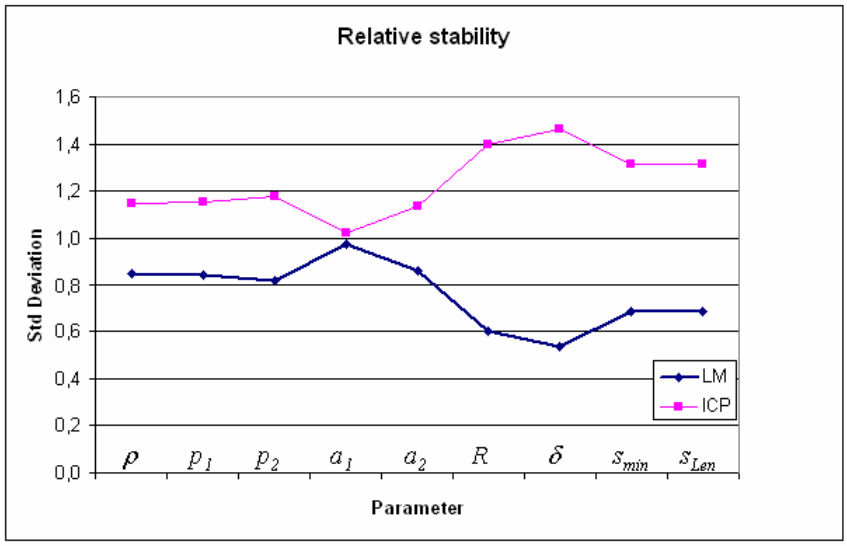

fig. 6-19 Relative stability of LM and ICP parameter optimization inside STD under all parameter variations

6.3 Evaluation of the local parameter optimization

Several experiments have been performed to answer questions regarding the EM algorithm:

- Within a single EM iteration loop, *several* algorithms optimizing slightly *different goal functions* work together in an alternating or interleaved mode: will they jointly converge, and under what conditions? Are the termination conditions mentioned in sub-section 5.5.2.4 meaningful?
- How fast do the fitting errors converge, and what about the estimated model parameters? How much accuracy is lost by setting data independent upper bounds on the number of iterations, enforcing termination and deterministic running times?
- As to the M-step, in particular, how do the Levenberg-Marquardt (LM) and the Iterative closest point (ICP) perform and compare?
- How do the configuration parameters, predominantly weight factors and thresholds in the E-step, influence the performance?

6.3.1 Termination conditions and convergence

Recalling section 5.5, each EM loop executes the parameter optimizer (M-step) once or several times using the same data points, and then calls the E-step to calculate probabilities to re-assign points to model primitives. Four values characterize this process: the root mean square of the fitting error *before* and *after* the M-step (called *PrevError* and *PostError*), and the quality q and weight w reflecting the mean respectively total degrees of membership assigned to points in the E-step. These four values as a function of iteration number were recorded using over 20 THERESA range views and different regions. In the so-called eqw-diagrams (error-quality-weight, figures 6-20), the units of measurement for *PrevError* and *PostError* are distances in mm. The dimensionless quantities q and w have been *scaled* to show them in a single drawing; weight w is shown at a logarithmic scale, and quality value q ($0 \le q < 1$) has been transformed by the function $(-\log_{10}(1-q))$ to better illustrate or 'zoom in' its residual growth as it approaches the maximum value of one.

Within the M-step, the selected fitting algorithm LM or ICP *decreases* the error. ICP, strictly speaking, guarantees monotonic convergence only if the closest point sets do not change and the geometric distance to the SoR surface is exactly evaluated and minimized. Then the *PostError* always stays below the *PrevError*.

Within the E-step, the support set changes; therefore, the mean fitting error (*PrevError*) may *increase*. Theoretically, there is no guarantee of convergence. Even an oscillating behavior with alternating growing and shrinking of the support set and corresponding increase and decrease of the error might occur. In practice and for a broad range of configurations, all true-positive cases however show a similar pattern seen in the diagrams in figure 6-20: after a fast growth in the first two iterations, the support set - its size correlates with the weight w- quickly settles down. Beyond the fifth or sixth iteration, only few scattered points are added or removed. Therefore, with a short delay, the fitting error (*PrevError*) ceases to grow during E-steps, and both errors converge to the same stationary value depending on the amount of noise in the data. I.e. in practice, the algorithm values converge.

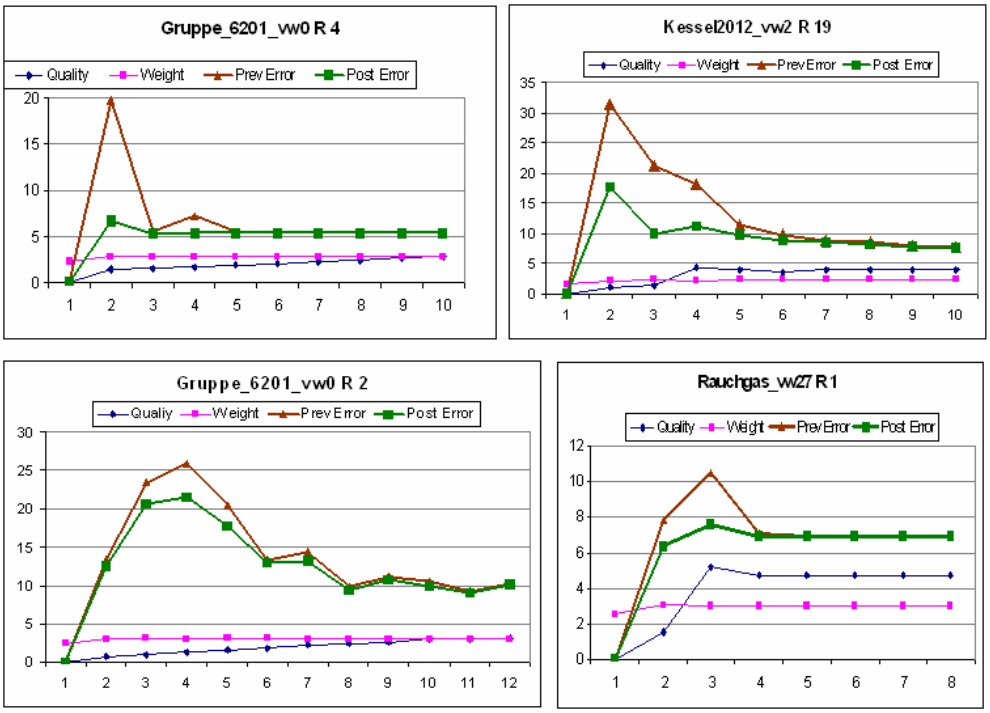

fig. 6-20 *Error-quality-weight* (*eqw*) diagrams of the EM state as a function of iteration step number; see text for explanations.

Therefore, a reasonable necessary condition imposed to continue iterating is a noticeable *increase* in hypothesis quality or in weight, *or* a corresponding *decrease* of *PostError*. A change by at least 0.1%, for example, is demanded. In all examples producing valid models, no more significant changes were observed after 10-15 iterations; thus, the EM algorithm always stopped owing to its own criteria. An additional *hard* upper bound on the number of iterations (set to 12) made bounded computation times certain. This *forceful termination* of EM ends any pathological indeterminate behavior, without sacrificing estimation accuracy in the true positive cases in practice. This shows empirically that forceful termination is justified.

6.3.2 LM and ICP comparison

Generally, the preceding observations apply to both algorithms, LM or ICP. Running on the same data, both methods achieve comparable convergence speeds and end up with comparable fitting error, quality, and weight values. LM seems to adjust the parameters slightly more 'aggressively' as shown in figure 6-21. On the other hand, ICP has a slight edge in stability according to figure 6-19.

Despite similar *eqw* diagrams, the estimated *model parameters*, e.g. the axis direction, may however differ. For example, LM occasionally under-estimates the opening angle on *conical* surfaces where ICP does not, such as the flue gas washer in figure 6-22. The initial hypothesis generation estimates an opening angle of ≈25°, close to the true value of ≈30°. While ICP does not change the angle value by more than 3°, LM sometimes *decreases* it from $\delta \approx 30°$ to ≈12°, and 'compensates' this by skewing the axis accordingly as seen in the parameter plots in figure 6-23. This happens between iterations 2 and 6; thereafter, LM is locked in the new local minimum.

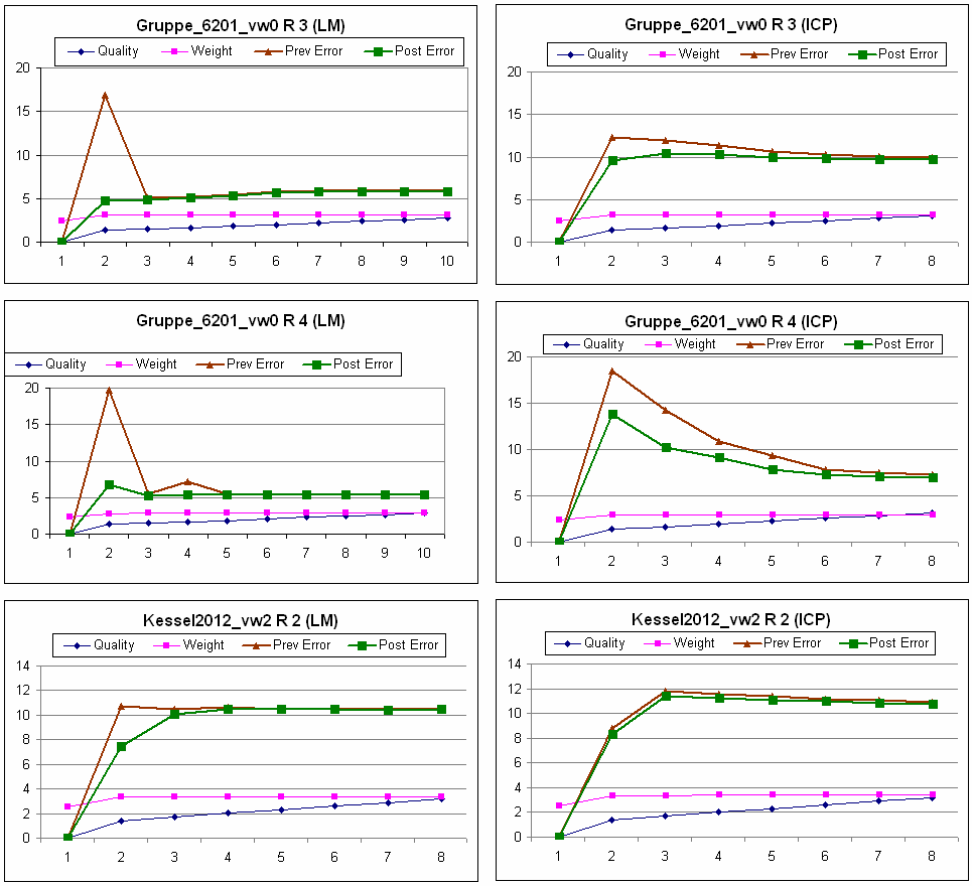

fig. 6-21 Behavior of LM (left) and ICP (right) on three example regions.

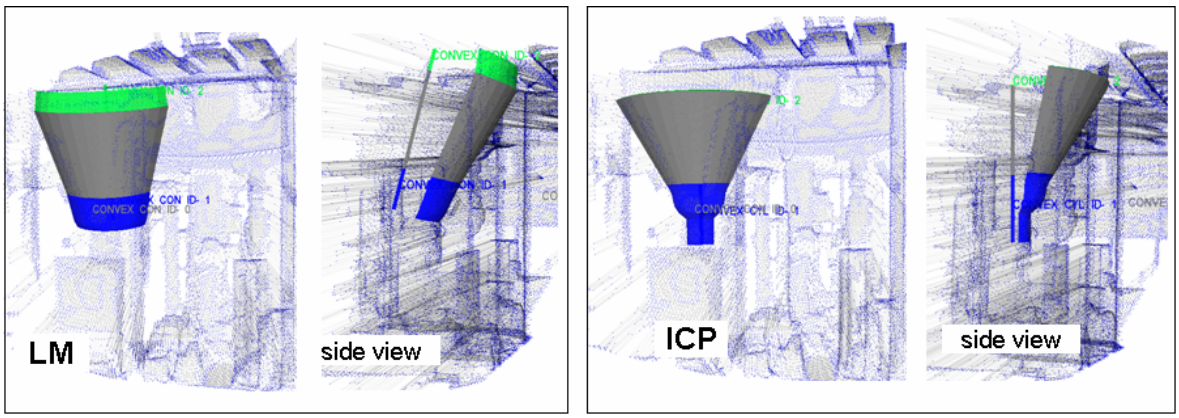

fig. 6-22 Example (Rauchgas_vw27) where LM skews the axis but not ICP

This transient change of parameter values is not detectable from the *eqw* plots[14]. It has neither been observed on cylinders nor on SoR with a 'flat' radius function. Not every one of six similar views of the same object and not all samplings were equally affected. Nor does the

[14] Notice also the reversal of the axis direction in the second iteration and the according sign change of the opening angle. This is not unusual to happen and no sign of an error. The conversion routines to and from Marshall representation were carefully checked for possible singularities and angular range problems.

problem occur *only* with the LM algorithm, but it does so more often (≈30% of runs) than with ICP (<10%). According to figure 6-22 ICP estimates the axis direction and also the opening angle accurately within 1-2°, as shown by the lower coaxial outlet. On the other hand, EM seems unable to fit a coaxial upper rim to the data for this choice of axis direction.

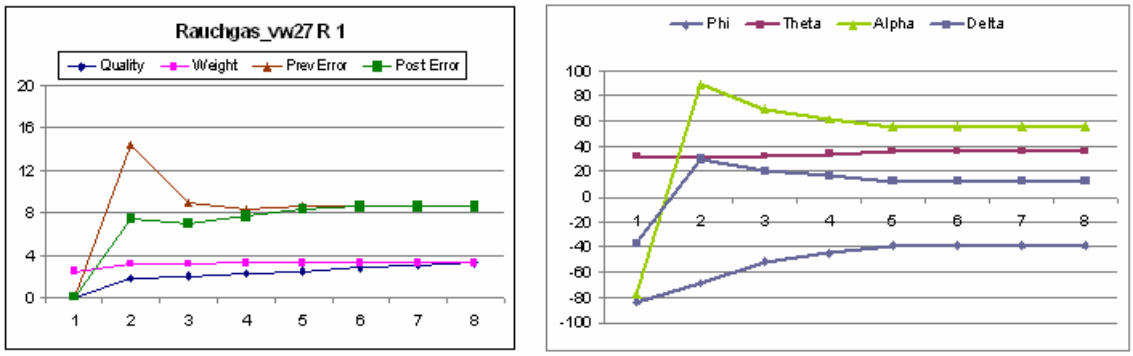

fig. 6-23 Using LM on the flue gas washer (Rauchgas_vw27): the fitting error settles rapidly according to the *eqw*-diagram (left) while some parameters keep changing, e.g. the opening angle δ from ≈30° to ≈13° (right).

The geometric distance measure itself cannot explain the differences as ICP and LM use the same *non-faithful* version, and numerical inaccuracies reported in [MLM01] could not be identified as a possible cause in these examples. A key difference between LM and ICP lies somewhere else: ICP itself, by estimating rigid motion, *only* acts on pose (axis) parameters; the shape parameters (radius function) are adjusted *separately* in the EM loop. Assuming that the initial model guess fits the seed region well, ICP will barely move the axis, causing only minor changes in the radius function update, on its part. This mutual tie is an advantage if both the axis and the radius estimates are good and an obstacle if both are poor. ICP cannot fix initial shape errors larger than 5-10° for the opening angle. In contrast, LM changes axis and radius parameters *together*; it may cause large changes in both of them with a *counteracting* effect on the fitting error.

Locating the axis direction within a profile plane is ill-conditioned using only the geometric point distance as a fitting error. Other error criteria, such as the *normal-ray-to-axis* distance used in hypothesis *generation* are not minimized in parameter *optimization*, the reason being the considerable computational overhead to calculate gradients of an extended error function. In the particular washer example the Plücker solution from normal rays (section 5.4.4) gives good and stable initial axis estimates needing little improvement, anyway.

Yet, we do not advocate settling for the coarse initial hypotheses. Direction estimates are often vulnerable to small normal variations, such as in the *Entspanner* (pressure reducer unit) images. High sensitivity occurs when the Plücker covariance matrix possesses a *cluster* of smallest eigenvalues (multiple ones, in the limit) spanning a subspace from which an eigenvector direction is picked. In the future, *sensitivity assessment* by the local optimization algorithm itself is needed at run time. Large directional parameter uncertainty despite small uncertainty of the fitting errors or matrix coefficients must be detected and analyzed for each data set. The parameter uncertainty is expected proportional to the inverse of the difference between the smallest eigenvalues of the (6x6 or 7x7) PCA matrix.

6.3.3 E-step configuration parameters

The optimization algorithm, in particular the expectation step, has several internal configuration parameters that influence its performance.

DoM function shape: In the actual E-step implementation, the normally distributed measurement probabilities, *exponential* functions of distances, are replaced by *trapezoidal* approximations, or *degrees of membership*. Such a DoM function has two parameters: a lower distance bound d_1 tolerated without any decrease, and an upper bound $d_0 > d_1$ above which the probability is zeroed. To further increase run time efficiency, points are discarded as soon as their overall probability, multiplying several DoM factors <1, falls below an acceptance threshold value th_{dom}. Two opposite configuration settings are compared:

1. A high acceptance threshold th_{dom} combined with a low discarding bound d_0 but a large toleration bound d_1 ('steep' configuration) and

2. A low acceptance threshold th_{dom} combined with a large discarding bound d_0 but a low toleration bound d_1 penalizing even small distances ('smooth' configuration).

The steep configuration may terminate weak hypotheses early which get a chance to converge under the smooth configuration. Slightly more false negatives therefore occur with the steep than with the smooth configuration. In some cases the convergence speed is slower with the smooth configuration. The stationary *PostError* values are similar under both configurations (figure 6-24). In general, the differences in fitting performance were much less significant than expected.

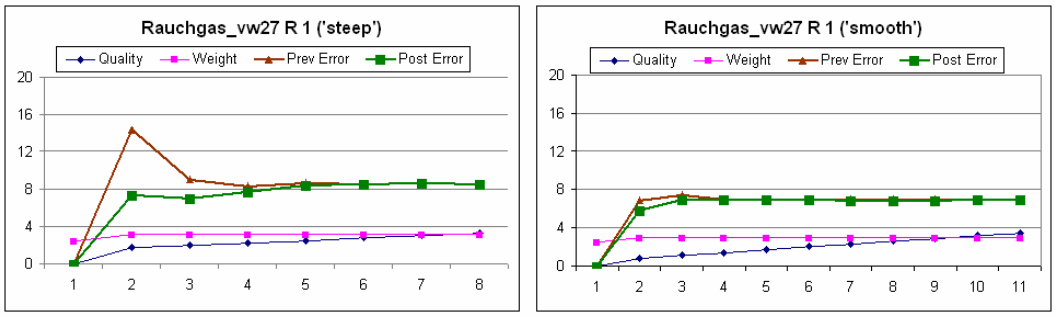

fig. 6-24 EM processing Rauchgas_vw27 using the steep (left) or smooth (right) E-step configuration.

Normal angle weight: The angular deviation of a point normal from the corresponding normal of the SoR primitive may either get a fixed importance (weight between 0 and 1) with respect to the geometric distance, or receive a variable weight *declining* with the iteration number. Our results indicate that a constant weight - be it high or low - implies fast termination and an annealing one slow termination. This relationship appears to be strongest for the smooth configuration where even small angle errors are penalized. But it may be due to an implementation problem: our DoM terms for geometric distance, normals, and projections currently do not get *individually normalized*, and the angle values are *lower* than the distance values. Therefore, with an *annealing* normal weight, the mean total DoM values keep growing even if neither points nor model parameters change.

Curvature type consistency: The E-step algorithm may 'trust' the initial curvature types attached by pre-processing and demand that they stay consistent with the assigned model primitive, or it may ignore point curvature types altogether. In cases when the starting parameter values come close to the final ones, little difference between both variants was observed, for example for the tank region (nr. 2, Kessel2012_vw2) shown in figure 6-25 on top. On badly segmented components with poor initial axis estimates, enforcing curvature consistency may help converge to the correct parameter values, for example for the pump-and-motor region (nr. 19) in figure 6-25 on the bottom.

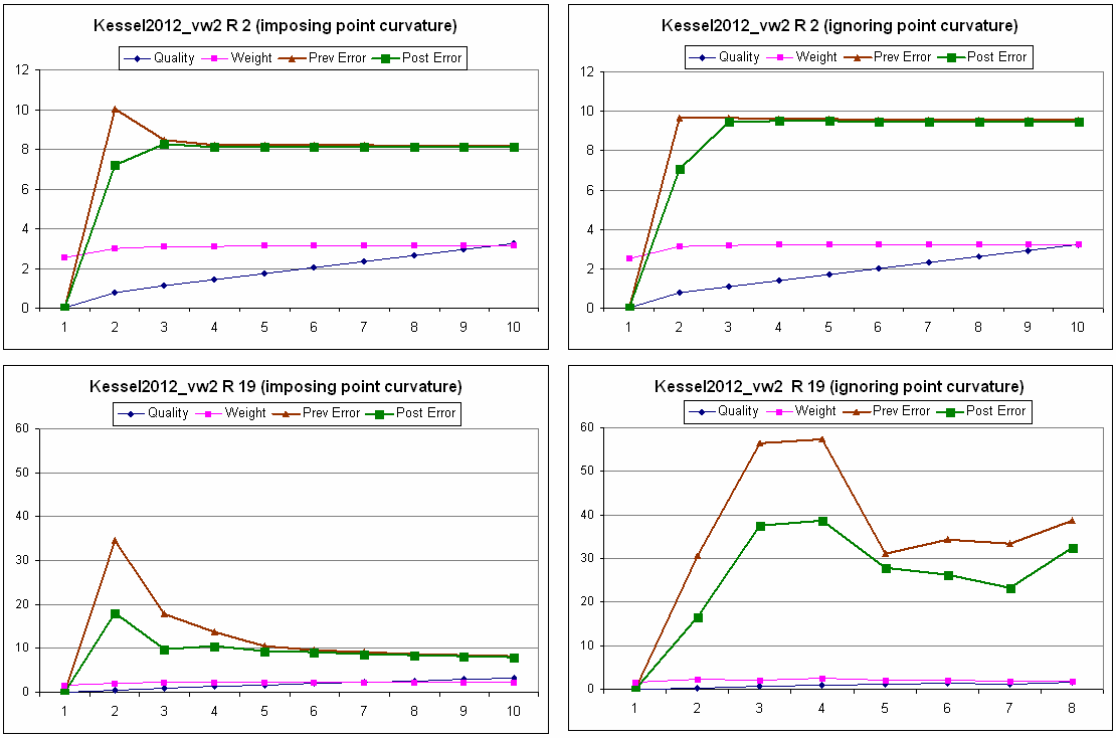

fig. 6-25 Examples of imposing (left) or ignoring (right) the correct point curvature types in the E-step.

Longitudinal growth ('greediness' parameter): This parameter determines if and to what degree a SoR hypothesis may grow beyond its initial *bounds* of *projection* onto the axis. It has a dramatic influence on the fitting errors and on convergence. If a hypothesis may aggressively extend its projection bounds, the resulting gain in *weight* may over-compensate the loss in *quality* and may prevent the fitting error from becoming stationary, as seen in figure 6-26 showing region 3 of the image Gruppe_6201_vw0. On the other hand, no ability to grow constricts the objects to their somewhat arbitrary seed region sizes. A good compromise suggested by the experiments is to allow limited growing during the first few iteration loops (say, by a factor of 0.1 during the first three loops) and stop growing as the seed region which has become sparse by sub-sampling has filled up again and become fully connected, mostly after 2-3 iterations.

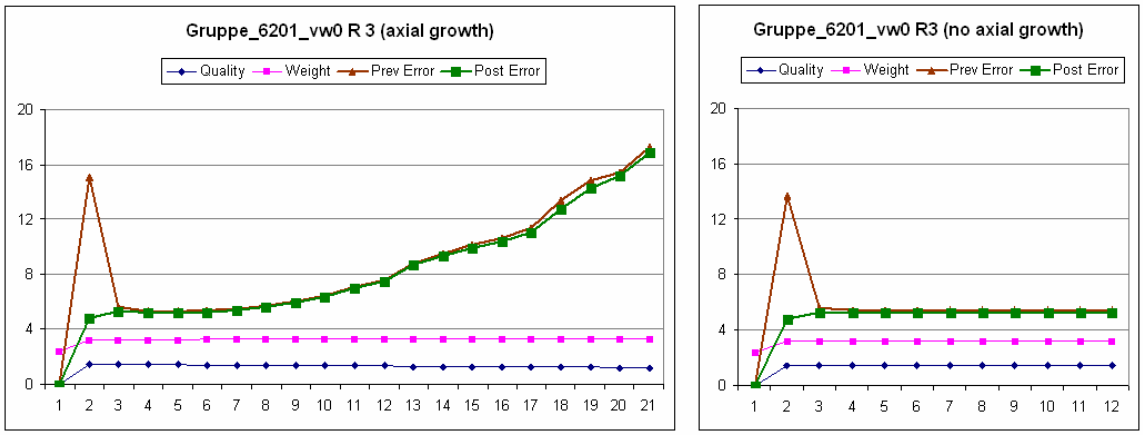

fig. 6-26 Growing in the axis direction may cause a diverging fitting error (left) which does not occur if no axial growth or limited growth is allowed.

6.4 Running time measurements

Timing measurements have been performed on a dual-core 3.3 GHz PC to assess the run time behavior of different algorithms and algorithm stages. Operating the standard Windows system clock at a resolution of ≈60 Hz requires small functions to run several (10-50) times on the same data in order to measure their mean running times. Visual C++ / Studio 2008 were used as the implementation platform. Possible interference of higher-priority system overhead with the application (single-threaded release version) was not taken into account. No optimization of algorithm implementation has been done as yet, except where explicitly stated in this report. Therefore, the results only give qualitative clues and do not apply to a true real-time implementation.

Two main questions were addressed: do we keep pace with our specific 3D scanner, and, more generally and importantly, is the running time of the entire estimation pipeline linear in the image size or number of points, assuming that the *capturing* time would be linear for any scanning system? And to what degree are running times predictable and deterministic?

The test image suite comprised 36 range views from the THERESA plant belonging to the seven groups mentioned in section 6.1. Figure 6-27 lists means and standard deviations of running times for 11 functions allocated to five major function groups:

1. Image Capture and Pre-processing: Read the range view from file (1.1); do uniform angle interpolation and Median filtering (1.2).

2. Point feature extraction: For all points, estimate the normal vectors and the strength of jump and crease edges (2.1); Estimate principal curvature values, types or shape descriptor values (2.2); Extract and label seed regions of homogeneous curvature type or shape; eliminate approximately planar components from further treatment (2.3).

3. Hypothesis Generation: Perform 6D/7D PCA analyses on the normal rays of all components suspected as SoR. Calculate rotational symmetry scores and decompose the point sets until straight-axis SoR pieces remain. Estimate the parameters for each segment, and sort the hypotheses by decreasing weight (3.1).

4. Parameter Optimization: Process the SoR hypotheses by decreasing priority using the EM algorithm (4.1). Evaluate the hypotheses and try a different algorithm (principal curve analysis on foot point clusters) where they do not adequately explain their component (4.2). Finally, decide for the best hypothesis (4.3).

5. Hypothesis verification: Evaluate the final hypothesis scores (5.1) and estimate the relations between the SoR segments (5.2). Discard all hypotheses failing the verification.

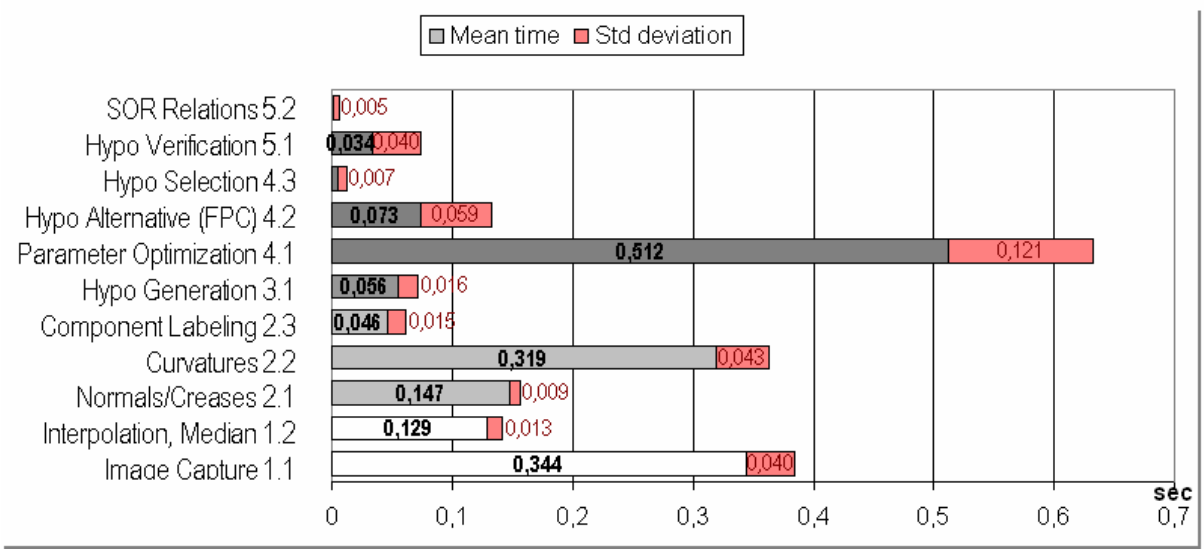

fig. 6-27 Running times of different functions

Running times of the group 1 functions are irrelevant as most of them would not be part of an actual plant mapping (SLAM) application. Reading the scene views from a file is replaced by scanning and streaming the image. The image capture times lie between ≈0.2sec and ≈5sec with our current *RoSi* scanner, depending on the *rotational velocity* set and on the scan angle resolution (100° scan angle at ¼° resolution, or 180° at 1° resolution). Obtaining a satisfactory radial resolution requires a rotational velocity so low that capturing a full image takes at least 1 sec. It is this 'frame' time the parameter estimation should keep pace with when speaking of 'real-time' mapping.

Interpolating the image to build a uniform angle grid is another feature peculiar to the experimental system. We do not recommend this step in a real SLAM system as it sacrifices one main advantage of the scanner: the excellent resolution in the image center. The core SoR estimation algorithms (function groups 3-5) do not assume uniform sampling density and require no interpolation. The main reason for interpolation was simplicity of normal estimation using a *sliding* window of *constant* size. A *spatially adaptive* window size would be needed to counteract the high sensitivity of normals in the image center where the radial sampling density tends to infinity. Interpolation supplies images of equal size (251x251=

63001 points). Median filtering might be useful in a real SLAM system to suppress isolated outlier points like mirror reflections; a naive implementation (no sliding window) and a small mask size (3x3) was used in the experiments.

The overhead of the function groups 2. to 5. is mainly due to point level features (group 2, 0.512sec in total) and to parameter optimization (group 4, mean 0.590sec), where EM has the largest share (0.512sec). Less than 10% of the total time is spent in hypothesis forming (group 3, 0.056sec) and hypothesis verification (group 5, 0.034sec). The methods chosen to form initial SoR hypotheses, especially the PCA methods using line geometry, are therefore efficient. The contribution of relation analysis to running times is almost negligible (≈2ms), although it is the only function having currently *quadratic* complexity, but quadratic in the number of SoR primitives.

Within group 2, the final component labeling step requires least time (0.046sec). Normal vector and edge strength estimation with sliding windows[15] take 0.147sec in the mean, and curvature estimation has the largest share (0.319sec). This is mainly due to the *principal curvatures* and *principal directions*; classifying curvature types or calculating local shape descriptors is negligible. Estimating principal directions at a sufficient angular resolution required at least 9x9 windows, and no sliding window operator provides minima and maxima in constant time.

The running times measured in groups 3 and 4 apply to the standard algorithm (STD). Corresponding total times for the Chain and FPT algorithms are: 0.439sec for the Chain, and 0.500sec for the FPT algorithm. Despite its slightly better timing results, the Chain algorithm, in particular, is not recommended for its poor stability and generally inferior performance.

Figure 6-27 shows also standard deviations of running times. For most functions, deviations relative to the means are moderate. Time dependency on the scene content seems to be bounded. The minimum and maximum running times of the EM algorithm amount to 0.266s for the *Kessel2012* scenes and 0.733s for the *Drehrohr* views, respectively. By forcefully terminating the EM, running times keep linearly bounded, proportional to the total number of points lying on rotational surfaces. The deviation of running times is larger for the optional FPC function. This comes at no surprise: the hypothesis generation from foot point clouds is invoked only under certain conditions and on a small fraction of badly segmented regions appearing with an unpredictable likelihood.

Since the EM algorithm accounts for more than 50% of total processing time and the M-step forms the dominant part, the influence of the fitting algorithm was analyzed in more detail (LM versus ICP, figure 6-28). Total running times of the EM are shown as the number of internal M-step iterations varies which are performed between two E-steps. Each data point represents an average from four types of scene with 10 randomized runs performed on each scene. The random strategy explains why the curves are not strictly increasing. The curves, at least for ICP, are not linearly growing but asymptotically flattening: adjusting the model

[15] Using a 9x9 window in all cases, the standard implementation (no sliding window, LINPACK eigenvalue routines) requires 10-15 times as much, i.e. 2-3 seconds (!) per image.

parameters to the previous E-step may actually require fewer repetitions than allocated. Five iterations are often sufficient for ICP to converge, but not so for LM. From the slopes of the initial linear parts of both curves one may estimate the mean overhead per ICP (≈90ms) and LM iteration (≈40ms).

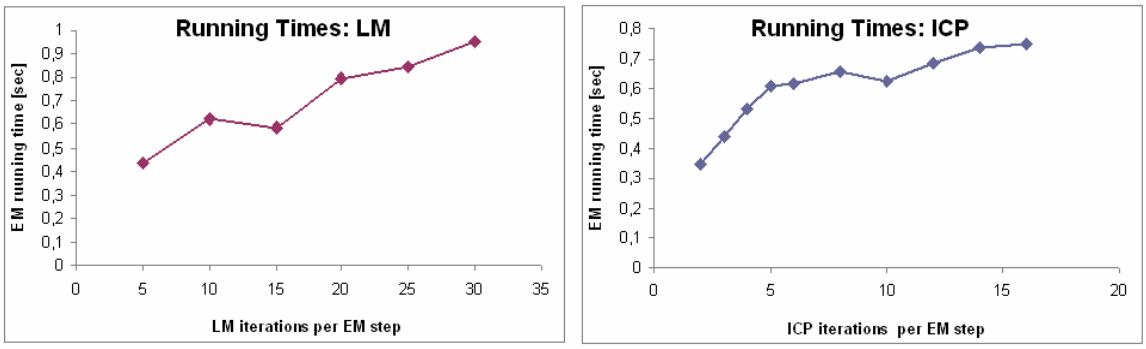

fig. 6-28 EM running time dependency on the maximum number of internal M-step iterations

The timing behavior when varying the number of combined E and M steps is shown in figure 6-29. Here, the number of internal iteration loops was fixed to some medium value (8 for ICP and 15 for LM). From figure 6-29 we see that the overall EM running times become *similar* for ICP and LM; at least, we cannot see statistically significant differences. This is surprising as the effects and the implementations of both methods are quite different.

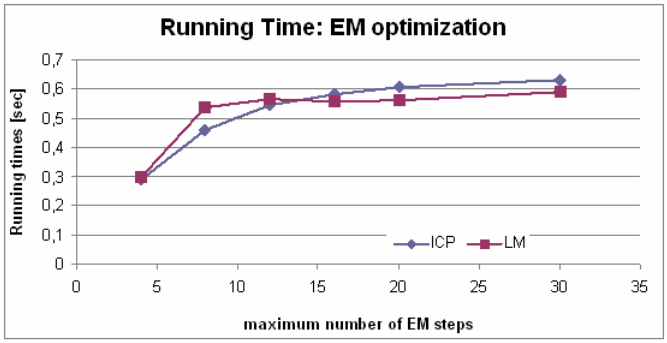

fig. 6-29 EM running time dependency on the maximum number of E- and M-steps combined

In figure 6-30, running times as a function of image size are plotted separately for group 2 (point features - dark blue) and for groups 3 and 4 (parameter estimation - pink). A collection of range views were interpolated at different resolutions; the image size on the abscissa includes only valid points and varies therefore. Fig. 6-30 shows two regression lines empirically confirming the linear relation between computation time and image size. This holds although parts, e.g. the Principal Curve algorithm (section 5.3.4) have above-linear expected complexity.

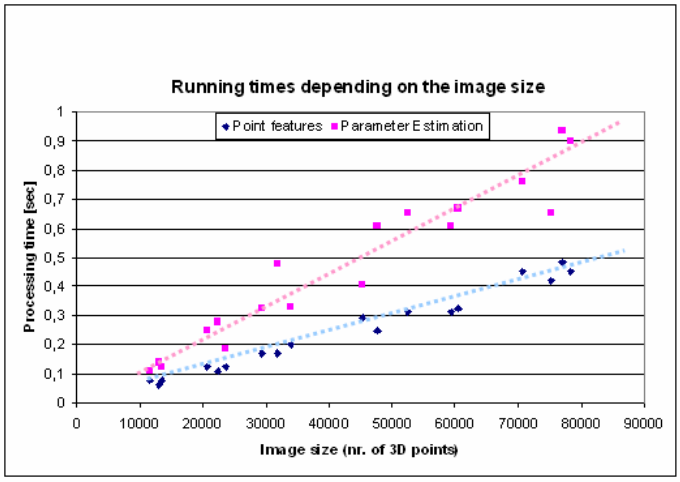

fig. 6-30 Running time dependency on the image size

6.5 Preliminary conclusions and discussion

As to hypothesis generation, the STD method *GenSORHypothesis* offered the best cost-benefit ratio in terms of efficiency, stability, and accuracy on those scene views showing clear and simple SoR structures such as linear axis and linear radius functions. Good performance is maintained on non-conical SoR or on surfaces with not perfectly circular cross sections, such as 'bruised' or facetted (prismatic) pipes and tanks. In rare cases, the foot point transformation with Principal Curve analysis, being inferior on the average, may be the only method achieving useable estimation results. These problem cases mainly occur if

1. the axis is a complex space curve, or

2. the visible shape has similar magnitudes of principal curvatures (spheroid, ellipsoid, or hyperboloid), i.e. the principal axis direction is hard to discern, or

3. the shape is heavily obscured, the spatial resolution is insufficient and the seed region provides no valid grouping information.

For the estimation in general, we see the following problems and limitations remaining:

- Thin objects resolved by few points (5-10) in cross section cannot be distinguished by their *cross section shape* (being circular, rectangular, T- or U-shaped), because of the high angular sensitivity. With the current laser scanner it is hard to discern thin pipes from slabs, beams, or strutting. This drawback cannot be overcome by estimation methods, only by a higher resolving scanner, or by an *active* (coarse-to-fine) *sensing strategy*, i.e. by moving closer to and by focusing on the objects of interest.

- Metric accuracy of the position of an *axis* curve including points of inflexion or branching will be poor when we see a Boolean union of rotation objects and the axes of symmetry are buried beneath complex blending functions. Part of the problem is inherently due to a low condition number as the visible patch covers a small angular section of the object. Clearly, the accuracy in these cases is far from being useful for reverse engineering applications. It remains to be tested if SLAM matching should at all use *large* rotation ob-

jects as landmarks. It is to note that SLAM pose estimation relies entirely on the estimated feature parameters, not on the raw data, and that errors accumulate over long view sequences.

- Very good starting values are crucial because the proposed parameter optimization (EM) cannot redeem a weak hypothesis, i.e. cannot escape from local optima of the fitting error, quality, or weight, plenty of which exist in the huge optimization space.
- Despite promising hypothesis evaluation and decimation a number of false-positive cones remain, especially 'hollow' cylinders simulated by concave crease edges. Arguing that concave cones occur much more rarely in process plants than convex ones, an obvious and easy-to-implement idea would be to penalize concave hypotheses by a low prior probability. Still, this would not completely solve the problem with creases.

We mention two more obvious ideas to lower the number of false positives:

1. Exploit the *lower stability* that hypotheses leading to the false positives tend to have. Create several *independent* hypotheses on the same sub-sampled region and assess their output similarity by the method of section 6.2. I.e., invest more time performing truly RANSAC (random consensus) hypothesis generation. A word of caution is in order though: even false hypotheses may attain a remarkable degree of stability and repeatability under random point selection. This action alone will not fix the problem.

2. Continue and toughen hypothesis evaluation, i.e. carry it on to the *multi-view* correspondence at SLAM level. Only true SoR hypotheses will appear in *several different views* at corresponding poses, and will therefore generate stable correspondences. We expect feature association to be a more effective place to decimate false positive features.

Our timing experiments indicate that the estimation algorithms are already suitable for real-time SLAM: their running times per image remain in the order of the capture time (1sec for ≈60000 3D points) and increase only linearly with the image size. Data dependency seems to be moderate, implying good predictability. A broader experimentation in different environment types is still needed to confirm this statement. Admitting that real-time performance may be easier to achieve than expected, the estimation results still leave plenty of room for improvement regarding accuracy and repeatability. The question arises if the computational effort is currently spent in the best way.

Point features

One may question whether *curvature information* is needed, and to what degree. Only the Chain algorithm and, to a lesser degree, the FPT algorithm require *directional* curvatures, neither the PCA based hypothesis generation nor the EM parameter optimization do. Partitioning into interest regions - considered necessary for performance as well as efficiency reasons - requires only coarse *integral* curvature measures, such as an approximation of the Mean curvature H, or levels of curvature magnitude, integrated over a region of points. Integral measures can always be calculated in constant time using sliding windows.

We feel that a large part of curvature estimation could be saved. In contrast, good *normal estimation* is considered essential to any competitive SoR estimation, and we provided a fast implementation for it.

Parameter estimation

The bulk of computation time is currently devoted to parameter *optimization* (EM), not to *forming* good hypotheses. When the results are unsatisfactory, poor *starting values* are the most likely cause. A point set not well representing the envisaged model primitive, and data noise boosted by an ill-conditioned estimation problem will quickly drive the EM to the closest local optimum. Trying to squeeze the lowest fitting error out of it does not pay, neither by spending more iteration loops, nor by using a faster converging nonlinear optimization routine, not even by choosing a different distance measure. For best use of resources, our experience suggests to perform the different modelling stages iteratively. After the initial hypothesis generation, the following steps may be repeated.

1. Evaluate the conditioning of the estimation problems, using the initial object parameters and support sets (sampled surface patches). A method similar to the Cramer-Rao bounds derived for circle fitting by Zelniker and Clarkson [ZeC04] might be generalized to 3D. If some hypotheses yield very large bounds, try to improve them by re-sampling the seed set or even redo the partition into seed regions. Otherwise, continue with step 2.

2. Run the EM parameter optimization for a few (2-3) iterations only, i.e. stop it prematurely.

3. Perform a preliminary hypothesis evaluation and relational analysis on the SoR elements obtained in 2.

4. Exploit the relations, in particular the parallel, co-axial, co-planar, and orthogonal ones revealing several SoR objects that form regular configurations. The condition numbers in step 1 indicate appropriate tolerance values for the relations to satisfy. Build large hypotheses from groups of smaller ones linked by the same relation; i.e. SoR sharing the same axis line or direction or lying in the same plane. If no more groups are found, continue with 2, or stop in case that the previous EM instance already 'converged'.

5. Make *compound hypotheses* from the relation groups, i.e. form the union of point sets and impose the relational constraints; for co-axial groups, no more constraints arise. Repeat the hypothesis generation step; i.e. estimate initial object parameter values for the compound hypotheses, and go back to 1.

This loop is driven by structural changes in the hypothesis set and by the *condition numbers* of the model parameters, not only by the *fitting errors*. As re-grouping settles down, gradually more effort may be invested in the fine optimization in 2.

Relational grouping in step 4 is an attempt to generate 'soft' constraint knowledge in man-made work spaces bottom-up from the data. It complements hard constraints explicit in a CAD model, such as the relations required between several pipe segments to form a T-junction [Rab06]. Clearly, when a point set is *known* to belong to a specific CAD object of this type, the knowledge should be used to get the best results. In plant exploration, such precise

knowledge is not granted. An example of possible coaxial grouping is shown in figure 6-31: the two objects were independently estimated with an orthogonal distance of ≈200mm between their axes, yet a relation of two coaxial adjacent rotation objects was generated with a confidence of ≈0.65.

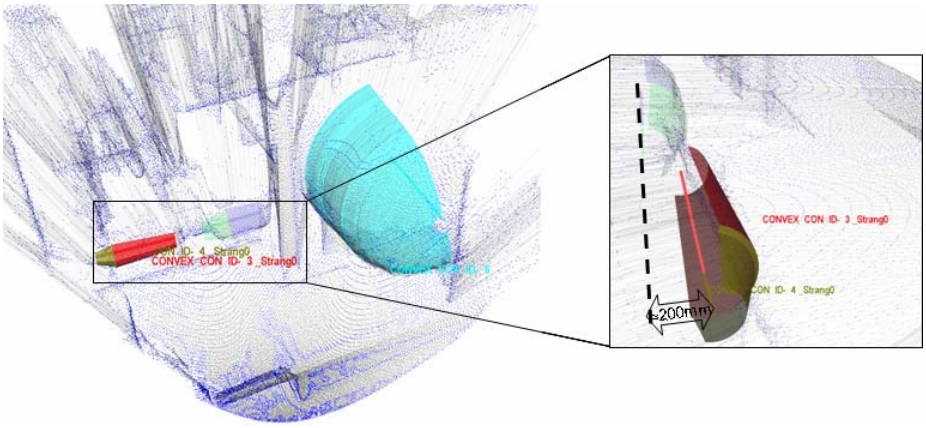

fig. 6-31 Detecion of co-axial SoR despite a large axis distance gives rise to new group hypotheses (range view: Kessel_2012_vw16).

7 Conclusions and future work

In this report a new tool box for the real-time parameter estimation of rotationally symmetric objects from range images has been developed and experimentally evaluated. The modeled objects include as special cases cylinders, cones, surfaces of revolution (SoR) with nonlinear radius function including ellipsoids as a special case, and ducts consisting of linked SoR with piecewise linear axis segments.

The algorithms form part of a feature-based SLAM system creating 3D maps of man-made work spaces from a stream of range views scanned by mobile observer platforms. Range views are raw 3D images with a given viewing pose and scanning order but without geometric structure. Compact parameterized features explaining the majority of data in buildings, settlements, and industrial plants include bounded planar patches and bounded solids of revolution. Other features with a few general pose and size attributes but no parametric representation may describe areas of clutter or complex free-form surface patches. The feature list will be extended; we expect it to gradually progress from a purely geometric towards a more functional description, discerning and denoting objects of special importance like slabs, beams, strutting, railways etc.

The goal is to build geometric part-structured and semantic data models of industrial legacy sites and to update existing CAD models using measured geometry data. The main purpose will be partly automated condition monitoring [KoB02] similar to the geo-referenced thermography [StH08]. Any infrared image captured should be automatically assigned to the right part and texture-mapped onto the part geometry. Thereby a part history ('anamnesis') is created that permits judging the part changes if the recording conditions are also clearly documented in the data model.

In this report we focused on solids of rotation and covered the entire estimation pipeline from raw range data to primitives: point feature and seed region extraction, hypothesis generation, optimization, and hypothesis evaluation including relation estimation. In each area suitable algorithms for real-time mapping were investigated. Generating hypotheses with good starting values of parameters and support sets turned out to be the key challenge. Among the options investigated, recursive region decomposition into linear axis segments based on line geometry based and guided by the degree of rotational symmetry, performed best on the average regarding reliability, accuracy, and efficiency. But also the Principal Curve decomposition of foot point clouds resulting from a foot point transformation yielded good results.

A new EM algorithm was developed for nonlinear parameter optimization. For parameter fitting in the M-step we choose between a Levenberg-Marquardt procedure, optimizing pose and shape parameters jointly, and a slightly more versatile Iterative Closest Point algorithm with separate shape estimation; both choices achieve similar performance levels. Several quantitative criteria were developed for hypothesis evaluation and decimation, for example the visibility (viewing sector), the degree of being tangential to other surfaces, and the surface coverage providing feedback as to when hypothesis generation might be redone using alternative algorithms. Relation estimation detects connected, co-axial, co-planar, parallel or branching SoR elements and also provides feedback to form new group hypotheses.

Extensive quantitative evaluation has been done on real imagery collected by a rotating laser scanner (*RoSi*) in an experimental plant designed for thermal waste treatment (THERESA). Running time measurements showed response times in the order of image capture times (≈1sec), and empirically confirmed that the computational complexity is essentially linear in the image size. In the course of stability experiments, covariance values were determined for the SoR parameters under typical noise conditions like random sampling, Gaussian noise, and motion noise. The covariance data are useful and are required to determine the uncertainty of the SLAM system estimating relative pose between range views.

Future work will go in several directions. Firstly, some more theory and implementation of *condition numbers* for SoR estimation from sampled surface patches is needed. Bounds on the covariance of SoR parameters (axis, radius) given an arbitrary patch sampled from the geometry should be explicitly expressed as a function of few parameters capturing the size and shape properties relevant for the estimation task. We expect the following parameters to be influential: one or two subtended angles measured from the center or axis of rotation, the angular sampling resolution, and an eigenvalue separation coefficient, measuring the difference between the principal curvature magnitudes or between the smallest eigenvalues of the 6x6 or 7x7 PCA matrix constructed from the Plücker coordinates. A formula reducing the input complexity would lend itself to on-line implementation and be highly useful to derive tolerance bounds, to gauge the uncertainty of SoR features as SLAM landmarks etc.

Secondly, the suggested feed-back loop from evaluation to hypothesis generation should be implemented and tested. Several solids of revolution linked by a common relation give rise to a new group hypothesis; the relation (being parallel, co-planar, parallel or co-axial) imposes additional and specific constraints on the SoR parameter estimation.

Thirdly, having successfully applied line geometry in a top-down approach, it seems worthwhile exploring also its bottom-up potential for SoR estimation, working on an 'ultra-coarse' seed region partition. Hand-tailored similarity measures of the PCA matrices from small regions, and how their properties change by merging need to be analyzed, similar to Guibas' approach [GeG04]. Besides similarity, conditioning seems to be as important.

Steps should also be taken towards the semantic or functional classification of important object classes occurring in industrial installations. Unsupervised learning algorithms appear especially convenient. New probabilistic methods for range images using Dirichlet distributions and Latent Dirichlet Allocation (LDA) have been recently proposed [EPS09]. We might clarify if the high mathematical and computational complexity would be justified in our application.

In the SLAM context, the SoR features need to be fully integrated into the data association and pose estimation algorithms. While much of this has already been done or is straightforward, one remaining task is setting up the similarity matrix for SoR relations. In previous work on building environments we derived benefits from orthogonal assemblies (stacks) of planar surfaces like walls, floors, and ceilings, leading to more reliable and efficient matching algorithms than e.g. interpretation trees (Orthogonal surface assignment [Koh07]). We might investigate if extending these concepts from planar features to rotation objects, such as pipe ducts, is worthwhile.

8 References

[AhE03] S.J. Ahn, I. Effenberger, S. Roth-Koch, E. Westkämper: Geometric Segmentation and Object Recognition in Unordered and Incomplete Point Cloud, 25th DAGM Symposium, Magdeburg, Germany, 2003, pp. 450-457

[Ang05] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng: Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data, Int. Conf. on Computer Vision and Pattern Recognition CVPR'05, 2005, 169-176

[ARC02] S. Ahn, W. Rauh, H. Cho, H.-J. Warnecke: Orthogonal distance fitting of implicit curves and surfaces, PAMI 24(5), 2002, 620-638

[BaF99] A.D. Bailey, C. Fröhlich: Reverse Engineering a process plant to produce a 3D CAD model, Numerisation 3D '99, Paris, France, Mai 1999

[BeF06] C. Beder, W. Förstner: Direct Solutions for Computing Cylinders from Minimal Sets of 3D Points, European Conference on Computer Vision (ECCV), 2006, 135-146

[BeJ88] P. Besl, R. Jain: Segmentation through variable-order surface fitting. IEEE PAMI 10(2), 1988, 167-192

[BeM92] P. Besl, N. McKay: A method for registration of 3-d shapes, PAMI 14(2), 1992, 239-256

[BRS03] T. Behrens, K. Rohr, H.S. Stiehl: Robust Segmentation of Tubular Structures in 3-D Medical Images by Parametric Object Detection and Tracking, IEEE Trans. Systems, Man, and Cybernetics, B33(4), 2003, 554-561

[CaF01] H. Cantzler, R.B. Fisher: Comparison of HK and SC curvature description methods, 3rd Int. Conf. on 3D Digital Imaging and Modeling (3DIM01), Montreal, Canada, 2001, 285-291

[CBP05] C. Colombo, A. Del Bimbo, F. Pernici: Metric 3D Reconstruction and Texture Acquisition of Surfaces of Revolution from a Single Uncalibrated View, IEEE PAMI 27(1), 2005, 99-114

[ChS92] X. Chen, F. Schmitt: Intrinsic surface properties from surface triangulation, Proc. of the European Conference on Computer Vision, 1992, 739-743

[CRD08] D. Cremers, M. Rousson, R. Deriche: A Review of Statistical Approaches to Level Set Segmentation: Integrating Color, Texture, Motion and Shape, Int. J. of Computer Vision 72(2), 195-215, 2007

[CSM07] N.D. Cornea, D. Silver, P. Min: Curve-Skeleton Properties, Applications and Algorithms, IEEE Trans. on Visualization and Computer Graphics vol. 13(3), 2007, 530-548

[DLP05] M. Deveau, G. Letellier, N. Paparoditis: Automating the extraction of revolution objects from single laser scans of architectural scenes, CIPA 2005 International Symposium, Torino, Italy, 2005, 746-749

[DoB03] P. Dokládal, I.Bloch, M. Couprie, D. Ruijters, R. Urtasun, L. Garnero: Topologically controlled segmentation of 3D magnetic resonance images of the head by using morphological operators, Pattern Recognition vol. 36, 2003, 2463–2478

[DRM07] A. Davison, I. Reid, N. Molton, O.Stasse: MonoSLAM: Real-Time Single Camera SLAM, IEEE PAMI 29(6), 2007, 1052-1067

[EPS09] F. Endres, C. Plagemann, C. Stachniss, W. Burgard: Unsupervised discovery of object classes from range data using latent Dirichlet allocation, Proc. Robotics Science and Systems Conf., Seattle, 2009

[FaF02] P. Faber, R. Fisher: Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting, Informatics Research Report EDI-INF-RR-0146, Univ. of Edinburgh, 2002

[FEF97] A. Fitzgibbon, D. Eggert, R. Fisher: High-level CAD Model Acquisition from Range Images, Computer-Aided Design 29(4), 1997, 321--330

[FlJ89] P. Flynn, A. Jain: On Reliable Curvature Estimation, Proc IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, Rosemont, Illinois (USA), 1989, 110-116

[FlJ91] P. Flynn, A. Jain: CAD-Based Computer Vision: From CAD Models to Relational Graphs, PAMI 13(2), 1991, 114-132

[Fre04] U. Frese: Ein O(log n) Algorithmus zur simultanen Lokalisierung und Kartierung von mobilen Robotern in Innenräumen, Dissertation, Universität Erlangen-Nürnberg, 2004

[Gar99] M. Garland: Quadric-Based Polygonal Surface Simplification, PhD dissertation CMU-CS-99-105, Carnegie Mellon University, USA, 1999

[GeG04] N. Gelfand, L. Guibas: Shape Segmentation Using Local Slippage Analysis, Eurographics Symposium on Geometric Processing, 2004, pp. 219-228.

[GFF08] J.-S. Gutmann, M. Fukuchi, M. Fujita: 3D Perception and Environment Map Generation for Humanoid Robot Navigation, Int. Journal of Robotics Research 27(10), pp. 1117-1134, 2008

[HaS89] T. Hastie, W. Stuetzle: Principal curves, J. Am. Stat. Assoc. 84, 502-516, 1989

[HoK06] A. Hornung, L. Kobbelt: Robust reconstruction of watertight 3d models from non-uniformly sampled point clouds without normal information, 4th Eurographics Symposium on Geometry Processing, 2006, 41-50

[HOP05] M. Hofer, B. Odehnal, H. Pottmann, T. Steiner, J. Wallner: 3d shape recognition and reconstruction based on line element geometry, Int. Conf. on computer vision (ICCV'05), 2005, 1532-1538

[Hor84] B.K.P. Horn: Extended Gaussian Images, Proc. IEEE 72(12), 1984, 1656-1678

[HSh03] E. Hameiri, I. Shimshoni: Estimating the Principal Curvatures and the Darboux Frame from Real 3-D Range Data. IEEE Trans. Systems, Man, and Cybernetics 33(4), 2003, 626-637

[HSi07] A. Harati, R. Siegwart: Orthogonal 3D-SLAM for Indoor Environments Using Right Angle Corners, 3rd European Conf. on Mobile Robots, Freiburg, 2007

[Jag05] A. Jagannathan: Segmentation and Recognition of 3D Point Clouds within Graph-theoretic and Thermodynamic Frameworks, PhD dissertation, Northeastern Univ. Boston, USA, 2005

[JiB97] X. Jiang, H. Bunke: Three-dimensional Computer Vision (in German), Springer Verlag 1997

[KeK00] B. Kégl, A. Krzyzak, T. Linder, K. Zeger: Learning and Design of Principal Curves, IEEE PAMI 22(3), 2000, 281-297

[KoB02] P. Kohlhepp, G. Bretthauer: Cooperative Service Robots for the Pedictive Maintenance of Process Plants, International Colloquium on Autonomous and Mobile Systems, Magdeburg, 25.-26.06.2002, pp. 53-59

[KoF99] P. Kohlhepp, D. Fischer, E. Hoffmann: Intrinsic Line Features and Contour Metric for Locating 3D Objects in Sparse, Segmented Range Images, Image and Vision Computing 17, 1999, 403-417

[Koh07] P. Kohlhepp, M. Strand, G. Bretthauer, R. Dillmann: The Elastic View Grah framework for autonomous, surface-based 3DSLAM, at-Automatisierungstechnik, Part I: Concept and Local Layer, pp.136-145, and Part II: Global Layer and Experiments, pp. 190-200, 2007

[Kop06] J. Kopp: Efficient numerical diagonalization of hermitian 3x3 matrices, arXiv:physics/0610206v1, 2006

[KPW04] P. Kohlhepp, P. Pozzo, M. Walther, R. Dillmann: Sequential 3D-SLAM for Mobile Action Planning, IEEE Conf. IROS, Sendai, Japan, 2004, 722-729

[KVD92] J. Koenderink, A.J. Van Doorn: Surface Shape and curvature scales, Image and Vision Computing,10,557-565, 1992

[Lei09] Leica Geosystems Homepage (Cyrax laser scanner, Cyclone software), http://hds.leica-geosystems.com/

[LeK07] F. Leymarie, B. Kimia: The Medial Scaffold of 3D Unorganized Point Clouds, IEEE PAMI 29(2), 313-330, 2007

[LiT94] P. Liang, C. Taubes: Orientation-based differential geometric representations for computer vision applications, IEEE PAMI 16(3), 1994, 249-258

[Liu01] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, S. Thrun: Using EM to Learn 3D Models of Indoor Environments with Mobile Robots, 18th Conf. on Machine Learning, Williams College, 2001

[LMM97] G. Lukács, A.D. Marshall, R.R. Martin: Geometric least-squares fitting of spheres, cylinders, cones and tori, Report GML 1997/5, Hungarian Academy of Sciences, Budapest, 1997

[LMM98] G. Lukács, A. Marshall and R. Martin: Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation, ECCV 98, Lecture Notes in Computer Science Vol.1406, Eds: H. Burkhardt and B. Neumann, Springer-Verlag, 1998, 671-686

[LoE06] G. Loy, J.-O. Eklundh: Detecting Symmetry and Symmetric Constellations of Features, European Conf. on Computer Vision (ECCV) 2006

[Lös05] M. Lösel: Unpublished Report (2. Praxissemester, in German), Forschungszentrum Karlsruhe, 2005

[Low04] D.G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints, Int. J. of Computer Vision 60(2), 2004, 91-110

[LTd90] P. Liang, J.S. Todhunter: Representation and recognition of surface shapes in range images: A differential geometry approach, Computer Vision, Graphics, and Image Processing 52 (1), 1990, 78-109

[LZK08] W.Li, A. Zhang, L. Kleeman: Bilateral Symmetry Detection for Real-time Robotics Applications, Int. Journal of Robotics Research 27, 2008, 785-814

[MLM01] A.D. Marshall, G. Lukács, R.R. Martin: Robust Segmentation of Primitives from Range Data in the Presence of Geometric Degeneracy, PAMI 23(3), 2001, 304-314

[NLH07] A. Nüchter, K. Lingemann, J. Hertzberg, H. Surmann: 6D SLAM - 3D mapping outdoor environments. J. Field Robotics, Special Issue Quantitative Performance Evaluation of Robotic and Intelligent Systems, 24(8/9), 2007, 699-722

[NuM88] A. Nutbourne, R. Martin: Differential Geometry Applied to Curve and Surface Design, Ellis Horwood Ltd, 1988

[OPW05] B. Odehnal, H. Pottmann, J. Wallner: Equiform kinematics and the geometry of line elements, Beiträge zur Algebra und Geometrie (Contributions to Algebra and Geometry) 47 (2), 2006, 567-582

[OrW00] X. Orriols, A. Willis, X. Binefa, D.B. Cooper: Bayesian Estimation of Axial Symmetries from Partial Data, a Generative Model Approach, CVC Report Nr. 49, Universitat Autonoma de Barcelona, 2000

[Pet02] S. Petitjean: A survey of methods for recovering quadrics in triangle meshes, ACM Computing Surveys 34(2), 2002, 211-262

[PoR98] H. Pottmann, T. Randrup: Rotational and helical surface approximation for reverse engineering. Computing 60, 1998, 307-322

[PrF88] W. Press, B. Flannery, S. Teukolsky, W. Vetterling: Numerical recipes in C: the art of scientific computing. Cambridge University Press, 1988

[PSG04] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, T. Funkhouser: A Planar-Reflective Symmetry Transform for 3D Shapes

[Rab06] T. Rabbani: Automatic Reconstruction of Industrial Installations Using Point Clouds and Images, Dissertation, TU Delft, The Netherlands, 2006

[Rab07] T. Rabbani, S. Dijkman, F. van den Heuvel, G. Vosselman: An integrated approach for modelling and global registration of point clouds, ISPRS Journal of Photogrammetry and Remote Sensing 61, 355-370, 2007

[ReV82] A. Requicha, H. Voelcker: Solid modeling: a historical summary and contemporary assessment, IEEE Computer Graphics and Applications 2(2), 1982, 9-24

[SaB93] H. Sato, T. Binford: Finding and Recovering SHGC Objects in an Edge Image, Computer Vision, Graphics, and Image Processing: Image Understanding 57, 1993, 346-358

[Sac02] R. Sacchi: Primitive-based segmentation for triangulated surfaces, PhD thesis, University of Nottingham, UK, 2002

[Sah06] K. Sahabi: Realisierung und Analyse eines FastSLAM-Algorithmus mit Flächenmerkmalen, Diplomarbeit (in German), Universität Karlsruhe, 2006

[Sco79] D. Scott: On optimal and data-based histograms, Biometrika 66(3), 1979, 605-610

[SD03] P. Steinhaus, R. Dillmann: Construction and modeling of the RoSi scanner for 3D range image acquisition (in German), Workshop on Autonomous Mobile Systems (AMS), Karlsruhe, 2003

[Ser82] J. Serra: Image Analysis and Mathematical Morphology, Academic Press (ISBN 0126372403), 1982

[ShH82] L. Shapiro, R. Haralick: Organization of Relational Models for Scene Analysis, IEEE Trans. PAMI 4(6), 1982, 595-602

[StH08] U. Stilla, L. Hoegner: IR-Texturing of 3D building models for thermal inspection of urban quarters (in German), GIS.Science 4, 2008, 12-19

[StR00] D. Stanford, A. Raftery: Finding Curvilinear Features in Spatial Point Patterns: Principal Curve Clustering with Noise, IEEE PAMI 22(6), 2000, 601-609

[SvB99] S. Svensson, G. Borgefors, I. Nyström: On Reversible Skeletonization Using Anchor-Points from Distance Transforms, J. Visual Communication and Image Representation vol. 10, 1999, 379-397

[SWK07] R. Schnabel, R. Wahl, R. Klein: Efficient RANSAC for Point-Cloud Shape Detection, Computer Graphics Forum 26 (2), 2007, pp. 214-226

[TaM99] C.-K. Tang, G. Medioni: Robust Estimation of Curvature Information from Noisy 3D Data for Shape Description, ICCV 1999, pp. 426-433

[Tau95] G. Taubin: Estimating the tensor of curvature of a surface from polyhedral approximation. Proc. 5th Int. Conf. on Computer Vision (ICCV), 1995

[Tay04] G. Taylor: Robust Perception and Control for Humanoid Robots in Unstructured Environments using Vision, Dissertation, Monash University, Clayton, Australia, 2004

[TBF06] S. Thrun, W. Burgard, D. Fox: Probabilistic Robotics, MIT press, 2006

[TCB08] N. Thacker, A. Clark, J. Barron, J. Beveridge, P. Courtney, W. Crum, V. Ramesh, Ch. Clark: Performance characterization in computer vision: A guide to best practices, Computer Vision and Image Understanding 109, 2008, 305-334

[TMK04] S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, E. Nebot: FastSLAM: An efficient solution to the simultaneous localization and mapping problem with unknown data association, Journal of Machine Learning, 2004

[TrF95] E. Trucco, R.B. Fisher: Experiments in Curvature-Based Segmentation of Range Data, IEEE PAMI 17(2), pp. 177-182, 1995

[Vin93] L. Vincent: Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms, IEEE Trans. Image Processing 2(2), 1993, 176-201

[Wel97] G. Welch, G. Bishop, SCAAT: Incremental tracking with incomplete information, Computer Graphics 31, 1997, 333–344

[WiC04] A. Willis, D.B. Cooper: Bayesian Assembly of 3D Axially Symmetric Shapes from Fragments, CVPR 2004, Washington DC, June 2004

[Wil04] A. Willis: Stochastic 3D Geometric Models for Classification, Deformation, and Estimation, PhD Dissertation, Brown University, 2004

[WoC08] W. Wong, A. Chung: Principal Curves to Extract Vessels in 3D Angiograms, IEEE Computer Vision and Pattern Recognition Workshops (CVPRW), 2008, 1-8

[XDQ04] H. Xie, K. McDonnell, H. Qin: Surface Reconstruction of Noisy and Defective Data Sets, IEEE Visualization 2004, Austin, USA, 2004, 259-266

[YaS03] F. Yao, G. Shao: Detection of 3D symmetry axis from fragments of a broken pottery bowl, International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03), 2003, 505-508

[ZeC04] E. Zelniker, I. Clarkson: Approximate Maximum Likelihood Estimation of Circle Parameters Using a Phase-Coded Kernel, 7th Eusipco Conf., Vienna, 2004, 617-620

[ZeN96] M. Zerroug, R. Nevatia: Three-Dimensional Descriptions Based on the Analysis of the Invariant and Quasi-Invariant Properties of Some Curved-Axis Generalized Cylinders, IEEE PAMI 18(3), 1996, 237-253

[ZoL06] H. Zou, Y. Lee: Skewed rotational symmetry detection from a 2D line drawing of a 3D polyhedral object, Computer-Aided Design 38 (2006), 1224-1232