SIXTH FRAMEWORK PROGRAMME
INFORMATION SOCIETY TECHNOLOGIES

robots@home

An Open Platform for Home Robotics

Specific Targeted Research or Innovation Project

Deliverable 3.1

Method of obtaining basic structural entities or parts

Date: May 5, 2008

Organisation name of lead partner: Vienna University of Technology, Automation and Control Institute

Contributors: Horst Wildenauer, Walter Wohlkinger, Peter Einrahmhof, Sven Olufs, Robert Vogl

Proposal/Contract no.: **FP6-2006-IST-6-045350**

Contents

1 Introduction and overview

Frequently occurring object parts or shapes are useful cues to help an autonomous agent to recognize and reason about the contents of a perceived scene. The key idea is to detect such patterns in a scene and together with learned configurations reason about the existence of an object(class) in the imaged scene. An argument for such an approach is rooted in the large intra-class variations of object's visual appearance which render holistic approaches inpractical. Here the assumption is that a decomposition in repeatable, frequently occurring patterns can levitate this hard problem to arrive at application-level relevant performances.

We want to stress that we do not want to restrict ourselves to a search for and interpretation of structural (physical) decompositions of objects, but rather try to identify frequently perceived patterns that are often shared between instances of an object class. In a very general setting, extracting the structural sub-parts of an object is hampered by the same problems as the recognition of an object itself, namely high intra-class variability. Eg., table or chair legs can occur in very different shapes, questioning the typical practise of approximating them with geometric primitives (eg., cylinders).

We choose to attack the hard problem of furniture recognition by the use of different sensor modalities which will later be combined to take advantage of their complementary nature while mitigating their unique disadvantages. Note that we deal with the problem of recognizing the class/category of viewed objects as opposed to identifying a specific instance (eg., recognizing a certain chair the robot has seen before). In past literature, the later was often termed object recognition, although mixed uses were and are very common. Note that in this report, when not stated otherwise explicitely, the term object recognition refers to the recognition of an object's class.

The present deliverable describes the characteristics of the sensor modalities in use, as well as the different approaches for the extraction of basic parts and entities of the robot's environment. Preliminary results are reported where available.

The rest of this paper is organized as follows: Sec. 2 describes the data sets collected during the first year of the project. In Sec. 3 we give a short introduction into the features and modalities used. Methods for the estimation of surface orientation and room layout are presented in Sec. 4. Sec. 5 concludes this report.

2 Data

The following sections give a description of the data sets which have been collected for the purpose of off-line testing and training. While the bulk of the data consists of material we have acquired using prototypical acquisition setups, we opted to include publicly available databases and collections of images from the WWW for the following reasons:

- Available ground truth and performance reports of state-of-the-art algorithms helps dissemination and comparison.
- Data unbiased by our own preferences w.r.t. data acquisition.

2.1 Wiry Object Recognition Database (WORD)

The Wiry Object Recognition Database (WORD) was created and is maintained by Owen Carmichael [Ch04]. According to Carmichael wiry objects are

> ... distinguished by a prevalence of very thin, elongated, stick-like components; examples include tables, chairs, bicycles, and desk lamps. They are difficult to recognize because their shapes are complex and they tend to lack distinctive color or texture characteristics.

Actually this characterization holds for many objects populating our everyday life, and as already mentioned especially for pieces of furniture like chairs and tables making this database highly relevant for the robots@home project. Of course, there are exceptions, such as eg. sofas and armchairs, which often sport very distinct patterns on their upholstery. Howsoever, while such patterns are very discriminative features useful for recognizing a specific instance of an object (eg, the sofa I bought last year), their arbitrary nature renders them to be of little use for classification.

The WORD data set provides a benchmark for evaluating shape-based object recognition approaches, providing ground truth consisting either of binary (detected) edges or of polygonal regions mapping onto the objects. A summary of WORD's characteristics is given in 1, example images can be found in Sec. 6, Fig. 42.

Object	#images	characteristics
Chair	116	full revolution in floor plane
Red chair	40	small shape deviation from chair
Cart	174	full revolution in floor plane
Ladder	1159	arbitrary out of image plane, seven different environments
Bicycle	151	three different poses, illumination changes, articulation
Clutter A	116	Office-like background environment
Clutter B	139	Office-like background environment
Clutter C	26	Office-like background environment
Stool	8201	taken in different office environments

Table 1: Characteristics of the WORD database.

2.2 The LabelMe database

The LabelMe database and web-based tool for image annotation [RTMF08] is one of the largest publicly available collection of images with ground truth labels. It is part of the computer vision communities ongoing efforts to provide large amounts of labelled data facilitating supervised learning of object classes and quantitative evaluation.

The LabelMe database provides a web-based annotation tool which allows visitors to annotate objects with polylines and arbitrary word-tags (such as, eg. car, windshield, stool, wall, etc.). There is no enforcement of annotation guidelines - except the option for users to correct annotations of others when desired. While annotation can only be carried out online, the database can be downloaded and individual object databases can be compiled off-line using a set of MATLAB tools.

For December 21, 2006, the authors reported the database to consist of roughly 111000 annotated polygons on 11845 static pictures and 18524 sequence frames with at least one object annotated. Since that time the database has grown rapidly, currently holding at over 160000 images. A summery of objects of interest for the robots@home projects is given in Tab. 2. Example images together with their ground-truth masks are shown in Fig. 43.

Object/scene	#images	notes
Chair	3692	indoor and outdoor images mixed
Sofa	394	indoor
Stool	71	indoor
Table	2924	indoor and outdoor images mixed
Cupboard	611	indoor
Door	3076	also contains car doors and gates
Window	11768	also contains car windows & building windows viewed from outside
Room	845	various room types with partially labelled furniture

Table 2: Relevant object and scene categories of the LabelMe database.

2.3 Single view image collection 1

This image set has been acquired by Horst Wildenauer using Google image search. It consists of 1900 images showing pieces of furniture in typical domestic environments (eg., dinning rooms, living rooms, kitchens) or against homogenous background (promotional images). 2500 chairs were manually annotated by bounding boxes, using the Matlab annotation tool shipped with the Pascal Visual Object Classes Challenge 2007 kit(VOC) [EVGW+07]. Example images are shown in Figures 1 and 53.

Figure 1: Examples from the single view image collection 1.

2.4 Single view image collection 2

The second data collection has been created mainly for the purpose of testing the algorithms for room-layout estimation.

Acquisition setup For this *single view image collection*, we used a standard consumer digital camera. These images are high resolution images taken at a height of approximately 1.6 meters.

IKEA This image collection was created at the event together with the stereo sequences at our visit to IKEA. The collection includes images of sofas, chairs and tables together with complex real-world configurations like kitchens, living-rooms and children's rooms. The lighting includes tungsten light and daylight which was no problem for the standard consumer camera but for our industrial cameras (see 2.5). A bigger snapshot of the *single view image collection 2* can be found in the appendix at Figure 44.

Figure 2: Examples of ikea single view image collection.

User provided domestic environments At the third robots@home meeting in Limoges, we took the opportunity to visit the Legrand demo house and made some pictures for our *single view image collection*. Figure 3 gives an idea of the images taken at Legrand. These images are not only for testing our algorithms, but also for planning future visits with James at the demo house for reduction of setup time.

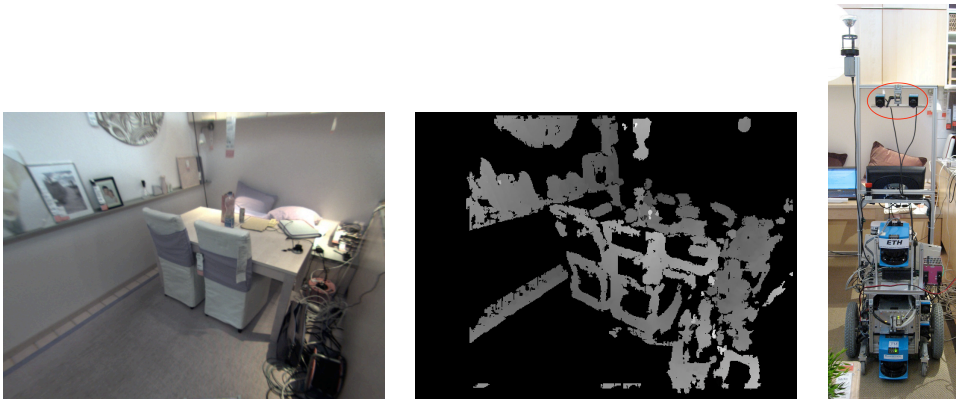
Figure 3: Example images of the legrand single view image collection.

2.5 Stereo sequences

As stated in the project proposal, several sensor modalities including stereo will be used in this project. In addition to the dense stereo data coming from the embedded stereo system, supplemental algorithms have to be tested for object avoidance, object recognition and object categorization.

Acquisition setup These stereo sequences were taken with a stereo setup consisting of a imaging source color camera (DBK 31AF03) and a imaging source monochrome camera (DMK 31AF03). The cameras were mounted on our robot at about 1.6 m above ground. The setup was calibrated before and the calibration data was saved together with the image data. To cope with the changing lighting environment - tungsten light and additional daylight through several windows on the ceiling - the cameras were configured to automatically adjust to the changing lighting conditions resulting in varying brightness along the image sequences. The absence of a IR-cut filter resulted in wrong colors in the color images. Our robot "James" with the stereo systems mounted on, can be seen in Figure 4(c).

The acquired data consists of a color image, a grayscale image, the configuration file and the computed disparity for the stereo setup. A typical stereo scene at IKEA can be seen in Figure 4(b) where the cameras look down to a table with chairs. As it can be seen easily on the disparity image, dense stereo is not the input modality of our choice for object categorization.



(a) Left image of the stereo setup. (b) Disparity of the scene. (c) James

Figure 4: Dense stereo and James.

IKEA The first field test for our robot was the two-days event at IKEA. One important task was to show the robot to the press, but the second task was much more important: Get as much data as possible. We recorded 167 Gigabyte of data. Most image sequences were taken by not autonomously driving around with the robot. A big snapshot of this huge amount of data can be found in the appendix in Figure 51.

To get test data which is as near as possible to reality, we also recorded sequences with the robot navigating autonomously. A snapshot of this data can also be found in the appendix at Figure 50. To give an idea of the living room the robot was driving in, Figure 5 shows a panoramic view of the environment.

Laboratory In the course of testing the object avoidance capability of the robot, we also recorded about 20 stereo sequences of moving towards furniture with the robot. The stereo system was also placed at a height of about 1.5m above ground. The furniture includes an old table, a chair and a couch chair which can be seen in Figure 6.

Figure 5: Panoramic view of the test-environment for autonomous data aquisition.

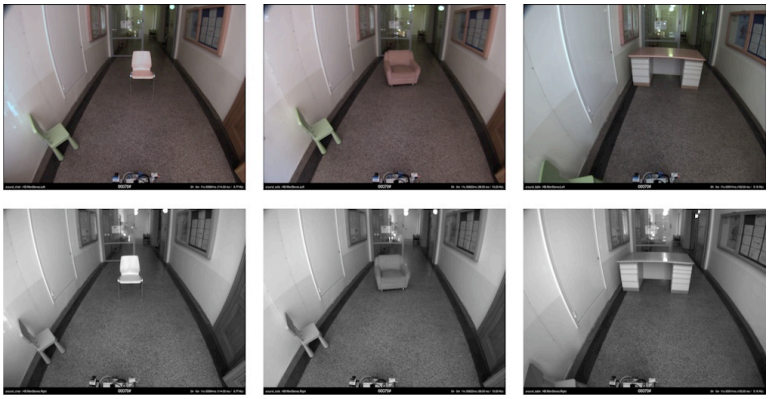


Figure 6: An example of the stereo sequences.

2.6 Trinocular image collection

Testing several algorithms and trying to get good results, we shortly found out that single view images and stereo images with a color and monochrome combination may maybe not be enough for finding the right method for obtaining basic structures for our task. Therefore, a setup was constructed which could deliver color stereo and trinocular images.

Acquisition Hardware The setup consists of three Imaging Source firewire cameras, two color cameras (DBK 31AF03) and one monochrome camera (DMK 31AF03), all with a resolution of 1024 by 768 pixel. As lenses we use three low-cost (about 20 each) 2.5mm mini s-mount lenses with a CS-2-C-Mount Adapter and C-2-S-Mount Adapter. The lenses have a horizontal field of view (fov) of about 140 degrees and a build-in IR-cut filter, which is a must-have, because our cameras are not equipped with an IR-cut filter.

Figure 7: A low-cost wide-angle S-mount lens with adapters from CS to C mount and from C to S mount.

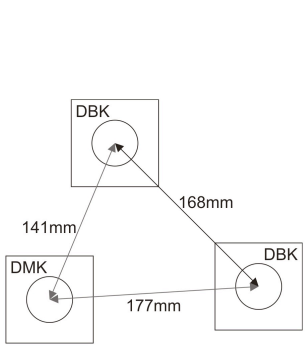
Acquisition Setup The configuration of the setup enables a dual-use of the sensors. At one hand it can be used as a standard trinocular setup and on the other hand, the two color color

cameras can be used as a standard stereo setup. The reasons and advantages of a trinocular setup will be explained later in section 4.3. The system was calibrated using a standard matlab toolbox [Bou07], but as this toolbox cannot calibrate trinocular setups, the multicamera self calibration tool from Tomas Svoboda [SMP05] was used. The arrangement of the cameras and the according baselines can be found in figure 8(a). As the final dimensions fo the robot and the final mounting of the camera setup is not known, we decided to test various scenarios. In Figure 8(b) the three different poses for acquiring the images can be seen:

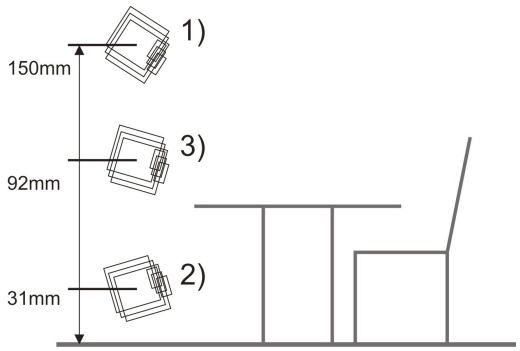
1. This is a good imaging setup with cameras at a height of about 1.5 meters which enables such a setup to look on a standard table (74cm height) with a top view (cameras tilted towards ground). This setup has another advantage: Most other image databases are also acquired from a standard human observer level, and can therefore easier be used. In general, acquiring data from a familiar pose and therefore having a priori knowledge of the environment, helps algorithms to perform better. [HEH06]

2. This setup mimics a small child. Cameras are at a height of about 31cm above ground. The cameras are tilted to look slightly upwards.

3. This setup was chosen to combine both advantages from setup (1) and (2): The height of about 92cm is just enough to look on a table and of course look downwards to chairs and sofas but without the need to create a mounting high above the robot. Mounting the cameras near to the wheels also reduces swinging and therefore enhances image quality.



(a) Arrangement of the three cameras. DBK = Bayer, DMK = Monochrome



(b) 3 different poses for image aquisition

Figure 8: Camera setup for Image Aquisition.

Laboratory image collection To provide a realistic test environment, we have created a living room in our laboratory, consisting of seven different chairs, three sofas and five tables sponsored by IKEA. Some additional furniture like book shelves, cupboards, desks and office chairs are also present. Figure 13 to Figure 15 show our test furniture in our laboratory. To give an idea of the quality and appearance of the images, figure 9 shows a sample scene of a chair taken with the setup described above. This *trinocular laboratory image collection* is used as test set in section 4.3.3.A bigger snapshot of the image collection can be found in the appendix at Figure 52.

(a) Left (b) Right (c) Gray

Figure 9: Images taken with the trinocular setup. The high radial distortion is apparent.

2.7 Range sensor data

For recording range sensor data we used the time-of-flight camera SwissRanger 3000 manufactured by the Swiss company Mesa Imaging[1] (Fig. 10). It has a resolution of 176x144 pixel, 25344 pixels in total. It should be noted that under the current circumstances this sensor is

Figure 10: Mesa Imaging's time-of-flight camera SwissRanger 3000

not ideal for robots@home due to these reasons:

- With a price of some 6000 Euros (exclusive taxes) for a single sensor it is too expensive for the desired low-cost robot system
- The field of view is rather small (47.5 x 39.6 degrees), so that the distance to objects has to be large (about 1.5m to 2.0m) in order to see the whole object or at least the bigger part of it. However, "typical" domestic environments are quite cluttered and often there's only little space to navigate
- The sensor requires active cooling (using a fan), which makes it noisy
- The SwissRanger is an active sensor. Two sensors of the same make interfere with each other. Besides, the emitted IR light's intensity is limited so that only short distances (a few meters) can be measured with sufficiently small uncertainty – the intensity of the light reflected by objects further away is low and therefor the measurement noise is high

Nevertheless, compared to normal cameras the SwissRanger has also advantages. Being an active sensor, it does not require a well-lit environment – it even works in complete darkness. Furthermore, there is no need for the environment to be textured as the camera measures the

[1] http://www.mesa-imaging.ch/

time the emitted IR light takes to travel to the objects and back again. Unlike stereovison, the SwissRanger works well on unichrome surfaces. The (uncalibrated) range measurement of each pixel is directly done by the sensor hardware, the computation of the calibrated 3D data – performed by the driver on a PC – requires only little CPU time. By lowering the integration time, framerates of up to 50fps can be achieved. To keep measurement noise low, framerates of around 20fps are more adviseable, though.

As mentioned above, currently the sensor is not suitable for letting the robots@home platform depend on it, but the technology [Kah07] is promising and and well worth investigating. Also, it has the potential to be low cost as the ZCam from 3DV Systems[2] demonstrates.

Figure 11: SwissRanger 3000 mounted onto a mobile robot. X and Z of the camera coordinate system are parallel to the floor plane. The center of the camera coordinate system is about 35 centimeters above the floor

Acquisition setup One of the recurrent questions in service robotics is where to mount the sensors for optimal results. On the one hand, sensor placement has to be suitable for the tasks of obstacle avoidance and self localisation, on the other hand there are constraints given by the size of the robot. For example, for a coffee-serving robot with a total height of fifty centimeters it's hardly feasible to mount a camera at a height of 1.3 meters. In the MOVEMENT project[3], the maximum height for mounting sensors was about forty centimeters to enable the mobile platform moving under a chair or table in order to dock with it.

As acquisition setup we have chosen a setup similar to MOVEMENT, with the X- and Z-axis of the camera coordinate system parallel to the floor plane and in a height of about 35 centimeters above the floor (Fig. 11).

Laboratory During the event at IKEA no data could be recorded with the SwissRanger 3000 as the time was too short to fully integrate this sensor into the acquisition software. Therefore, one office at ACIN was refurnished using IKEA furniture only, in order to create an in situ testing location (Fig. 12). Several series with 300 frames each were recorded while the mobile robot was moving through the office.

Every recorded frame comprises a timestamp, the raw (i.e. uncalibrated) range measurements, the amplitudes and the calibrated 3D data. Fig. 16 shows the visualisation of several example frames. The left column shows the amplitudes values. The raw data are 16bit unsigned integers – for better visibility, these have been squareroot-scaled and the resulting 8bit images have been normalised. The middle column shows the depth values, i.e. the Z-coordinate of the 3D data. Again, for better visibility the images have been normailsed. Finally, the right column shows the 3D data. The coordinates are represented in the form of signed 16bit integer triples, the unit is millimeters.

[2]http://www.3dvsystems.com/technology/tech.html

[3]European Union project MOVEMENT #IST-2003-511670

Figure 12: Office at ACIN, refurnished as in situ testing location

Figure 13: Test chairs from Ikea: Klapsta, Harry, Stefan, Ivar, Morits, Olle, Terje.

Figure 14: Test tables from Ikea: Ingo, Lack, Lack, Liden, Mikael.

Figure 15: Test sofas from Ikea: Lillberg, Klippan, Klobo.

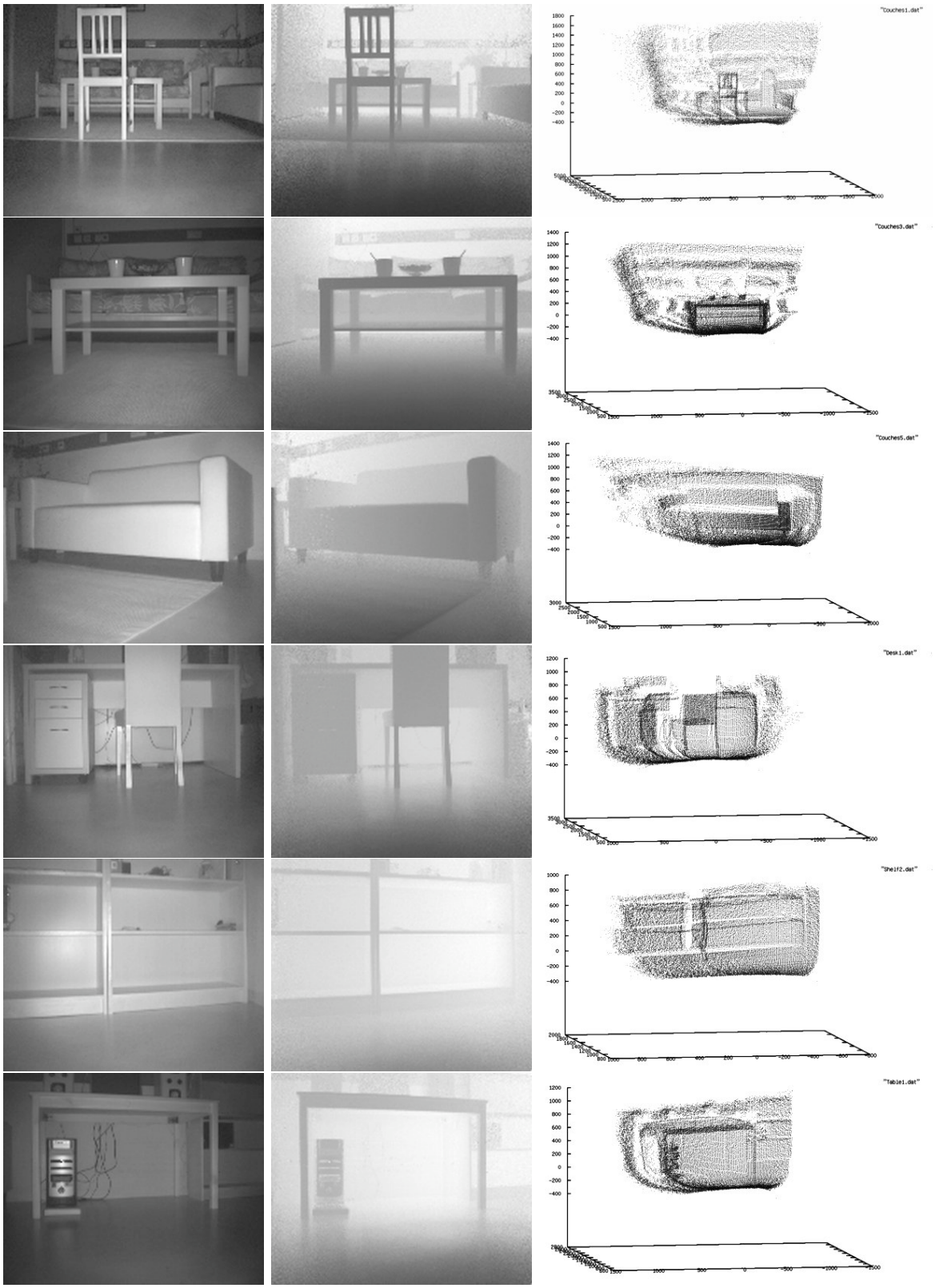


Figure 16: Several examples of amplitude, depth and 3D data recorded with the Swissranger 3000

3 Basic features and modalities for furniture recognition

3.1 Superpixels

The concept of superpixels was originally proposed by Xiaofeng Ren and Jitendra Malik in their seminal work on the learning of classification models for image segmentation [RM03]. Here, the key idea is to replace the pixel representation of an image by local, coherent, and structure preserving image regions. These are typically obtained by over-segmentations adopting the following considerations:

- Intra region properties
  - Brightness similarity
  - Texture similarity
  - Weak contours inside region
- Inter region properties
  - Brightness dissimilarity
  - Texture dissimilarity
  - Strong contours along the separating region boundary
- Curvilinear continuity
  - Boundary smoothness

On of the main advantages of superpixels is that they offer a low-complexity image representation while still trying to retain the information necessary (eg., respecting segment boundaries [RM03]) for further processing steps (such as, eg., image segmentation [MP07, WMV07], object classification [RES+06], geometric surface context estimation [HEH05, HSEH07]). For typical images the number of superpixels ranges in the range of hundreds to thousand, which is more then several orders of magnitudes lower than the average number of pixels. In contrast, simply reducing the image size and building on pixels to avoid complexity as implemented in many approaches leads to losing details and high texture frequencies.

In our work, see Sec. 4.2.3, we utilize the efficient Minimum-Spanning-Tree (MST) based colour segmenter described in [FH04]. This method facilitates Kruskal's algorithm on a (pixel) grid graph, merging regions based on a region-size to intra/inter-region colour similarity/dissimilarity criterion. Typical computation times are about 0.5 seconds for a 800x600x3 image, where a significant portion is consumed by pre-smoothing and the construction of the grid graph. Examples of the results obtained with this algorithm are given in Fig. 17. Note how the superpixel boundaries nicely follow object boundaries and surface discontinuities.

3.2 Finding straight line segments

Edge detection We compute pixel gradient strength and orientation using Gaussian derivatives, as suggested by Canny. Prior to this step, images with less then one Megapixels are up-sampled by factor two, which substantially increased the number and quality of lines segments detected in the whole process [Köt03]. Gradient computation is followed by adaptive

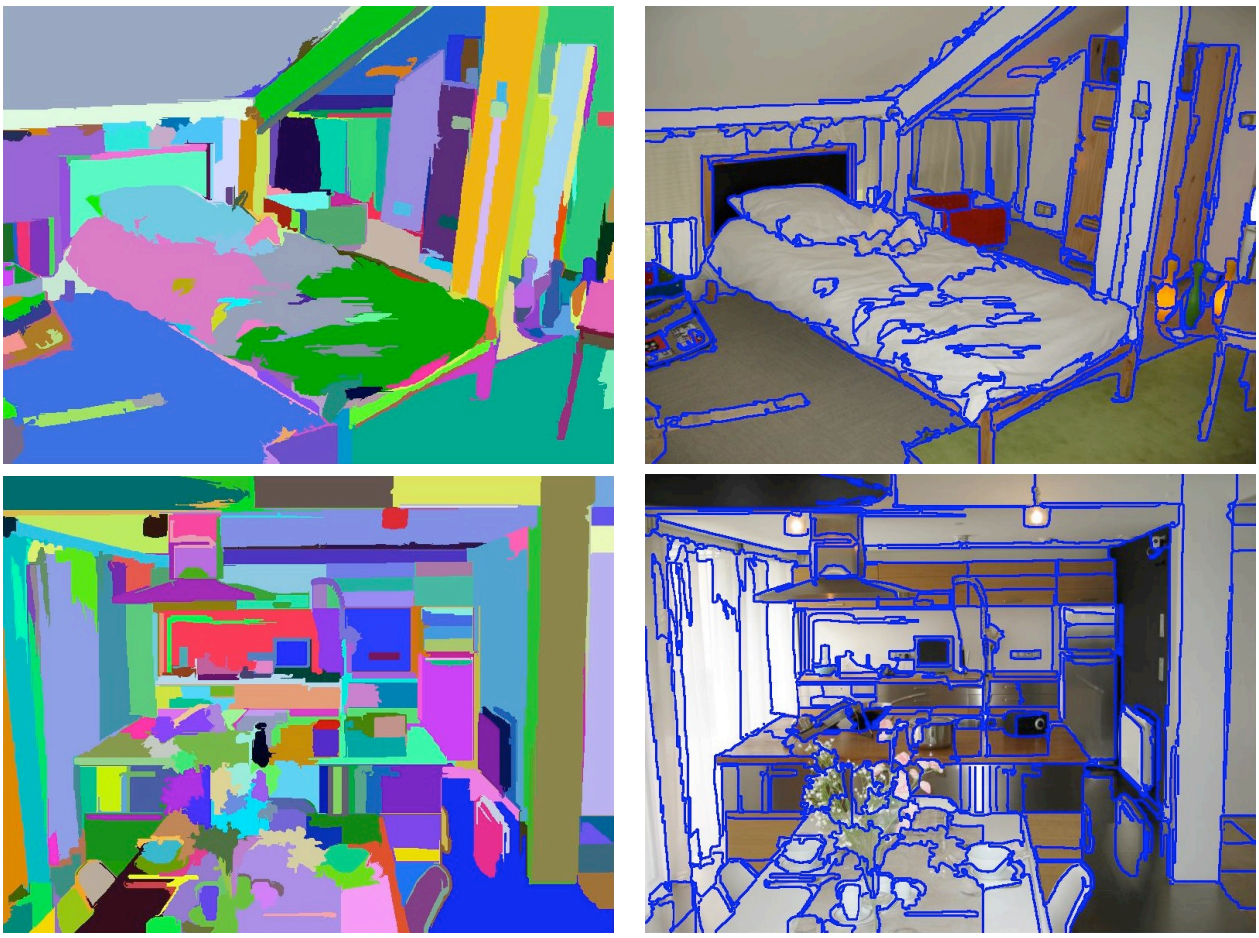


Figure 17: Superpixel over-segmentation of images taken at Legrand,s using the method of Felzenszwalb [FH04]. Superpixels detected in an image. Right: Original image with superpixel boundaries overlaid.

hysteresis-thresholding with a conservative upper threshold at 50% of the image gradient energy, and edge-linking with subpixel accurate non-maxima supression.

Due to the relative low adaptive thresholding on the gradient energy, the proposed approach produces overly rich populated edge images - increasing the amount of spurious edges considerably. This however has the advantage that relatively low contrast edges at surface orientation discontinuities (eg. the edge between two joining walls) that give rise to stable, long enough line segments are still accepted.

Line segment extraction For the extraction of straight segments from edges we perform the following steps

1. Splitting of linked edgels at junctions (detected by a simple and efficient morphologic operation) into separate edge segments.

2. Subdivision of linked edgels into approximately straight line segments using the iterative scheme, as implemented in Peter Kovesi's MATLab Toolbox [Kov].

3. Finally, line segment parameters (midpoint, endpoints, and orientation) are obtained by a total least squares (TLS) fit (see, e.g. [KZ02b] to the pixel coordinates of the edge

segments.

Finally, low quality line segments, i.e. those shorter than 20 pixels, or with a small eigenvalue of the pixel-coordinate covariance matrix larger 0.3 are rejected. For the majority of our test images, this step reduced the clutter considerably, as noisy edges are unlikely to be well approximated by long straight line segments.

Depending on the size of the image and the structure of the imaged scene, one typically obtains line segment numbers in the order of several hundreds. An example of the typical outcome of the line detection stage is shown in Figure 18.

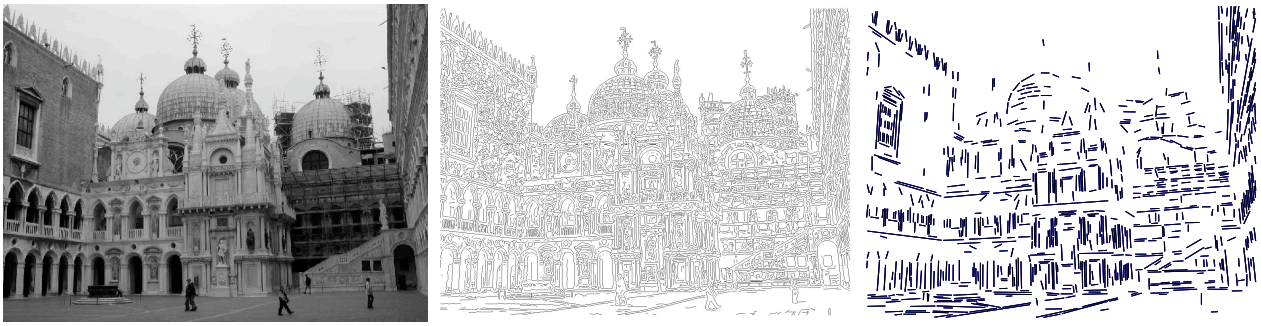


Figure 18: Example of the line detection process. Original image, edge image, and line segments.

Comments on the used edge splitting technique In literature a vast number of techniques for polygonalising curves exist and a thorough discussion is beyond the scope of this paper. However, the idea common to most of the methods is to detect high curvature points along a curve which are then used to split the curve into a piecewise linear approximation.

In our work, we adopt a simple and computationally efficient scheme similar to the one proposed by Lowe [Low87]. Splitting points are detected by finding the point on a curve with maximum orthogonal distance from a line segment connecting the endpoints of that curve, see Fig. 19. The splitting is iterated until no more points with a deviation above a predefined threshold (typically 1.5 - 3 pixels) are found.

In contrast to our method, Lowe and later Rosin and West [RW95] use a recursively defined normalizing measure of line segment significance to obtain a scale invariant curve polygonlisation (ignoring discretisation effects). In fact, this favourable property often drove researchers (eg. [Zil07] to use this method for straight line extraction. However, we found that its application did not give the expected benefits in our problem domain, as it only holds for closed curves which seldom occur in real images.

3.3 Dense stereo

The main problem in dense stereo to face is the acquisition of accurate disparity or depth data. While recent advances in stereo methods improved global matching and the ability to handle occlusions, these methods are time consuming and not feasible for mobile robot applications [Bro03]. Even under good circumstances, the accuracy of typical disparity (and depth) images contain rough or wavy surfaces with missing or spurious data points. With the increase of computing power of general purpose PCs, there are now methods available that produce disparity images at frame rate [Bro03]of $15Hz$. A typical disparity image of a scene

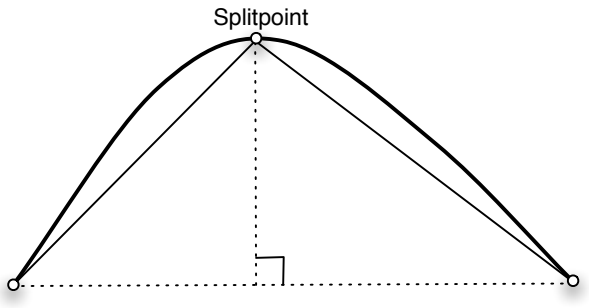
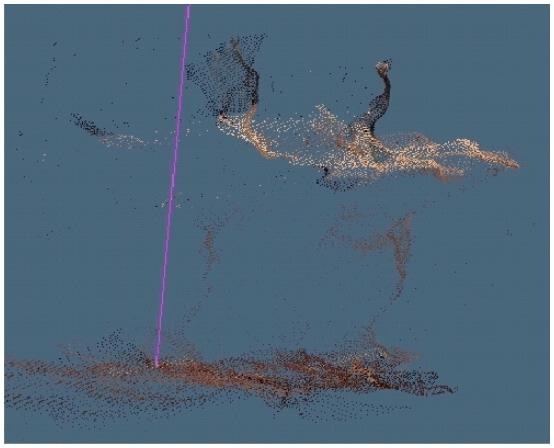


Figure 19: Approximating an edge by line segments. An edge is iteratively split at the point of maximum deviation from a line segment connecting its endpoints.

in a living room is given in Fig. 20(b). The wavy structure picked up due to relatively little or no texture can be seen clearly. However, most tables and floors exhibit little texture, such that mobile robotics needs to cope with this situation.

(a) A typical scene where laser range finders have difficulty to detect the table.



(b) A typical disparity image: the data points are viewed from the side. The purple, nearly vertical line is the estimated normal to the ground plane.

4 Estimation of geometric surfaces and room layout

In this section we describe techniques for planar surface detection and room layout estimation developed in the first year of the robots@home project. Specifically, we focused on: (a) Monocular cues for detecting homogeneous planar patches and their orientation, as well as camera orientation w.r.t. to a imaged room. (b) Plane extraction using a Trifocal tensor-based approach. (c) Dense stereo for ground plane estimation and simple furniture detection. Additionally, a brief glimpse at the possible application of range data-based methods, planned for the future course of the project, is given.

4.1 Camera orientation and line segment grouping from vanishing points

Man-made environments generally exhibit strong regularity in structure and often many parallel lines are present. In such settings, vanishing points and lines provide useful visual cues for deducing information about the 3D structure of the imaged scene. In fact, assuming a calibrated camera, with the detection of vanishing points and vanishing lines, the relative orientation of imaged lines and planes w.r.t. to the camera and vice versa is uniquely determined.

Furthermore, if two or more vanishing points are found of which the underlying structure's orientations are assumed to be orthogonal, then, taking mild assumptions [4] internal camera parameters can be determined by solving a set of linear equations. Three vanishing points allow for the estimation of the focal length and the principal point. Using two vanishing points only the focal length can be estimated (the principle point is assumed to be in the centre of the image)

As a consequence of these facts, the problem of reliably finding vanishing points has been addressed numerous times in the past. E.g., for the case of a calibrated camera, a preferred representation is the Gaussian sphere, used as accumulator space for Hough-based approaches [MA84, AT00, BO91].

Although a calibrated acquisition setup can be taken for granted on our experimental platform, we will also also deal with the uncalibrated case. This enables us to access the rich set of publicly available, annotated data collections, allowing for comparisons with state-of-the-art techniques, and helping to obtain performance assessments unbiased by our own preferences w.r.t. data acquisition.

In our work, we reconsider approaches that try to exploit the so-called Manhattan world assumption [CY03]. It is most closely related to the work of Kosecka and Zhang [KZ02b], who exploit orthogonality by grouping line segments in a hypothetical calibration setting. We use a refinement scheme similar to theirs, but show that dominant structures can even be found in complex settings, when the method is properly initialized and the grouping is carried out in a data-driven manner.

4.1.1 Finding vanishing points using line segments

Here, the idea is to repeatedly generate vanishing point hypotheses through the intersection of lines. Intersection points having a large enough set of lines pointing towards them, are likely to be true vanishing points and are reconsidered in further processing stages.

In our approach, we follow the work of Pflugfelder and Bischof [PB05], and Aguilera et al. [AGLF05] and use a RANSAC-based initialization scheme.

[4] Assuming zero skew and unit aspect ratio

RANSAC-based line clustering Since the actual mixture fraction of lines belonging to different vanishing points is unknown, we adopt the adaptive variant proposed in [HZ04]. Specifically, we run the algorithm several times over the dataset and successively remove the largest found inlier set from the data before the next trial. After each trial, the vanishing point position is refined by applying Kanatani's *renormalization* scheme [KS05] to the respective consensus set. We reject newly detected vanishing points if they lie within the uncertainty of previously detected ones utilizing the test statistics proposed in [PB05]. Here, however, we adopted the vanishing point covariance matrices obtained by *renormalization*. The iteration is stopped, if no more consensus sets with a cardinality above a predefined threshold are found, or when a predefined number of vanishing point candidates K_{max} is reached. We found that $K_{max} = 20$ gives a good balance between adequate exploration and computational cost for the following processing stages, and consequently used it all our experiments.

For sake of completeness, we want to mention that we do not utilize the χ^2 test statistics (described by Hartley in the same reference above), since we want to minimize the chance of removing a true outlier (which could be an inlier for an other vanishing point) from the data. Instead, a experimentally determined threshold was used.

Line error model To quantify the error of a line segment meeting a vanishing point, an ideal line from the segment's midpoint to the vanishing point is constructed and the normal distance of one segment endpoint to this line is measured. Formally, this distance can be written as $d^2(\mathbf{a}_i, \bar{\mathbf{a}}_i)$, where $\mathbf{a}_i$ is the measured line segment endpoint, and $\bar{\mathbf{a}}_i$ is its root point on the ideal line. Hence, shorter line segments are allowed to exhibit stronger angular deviations than longer ones.

The described model is based on the assumption that there is little variation in the midpoint of the line segment, as it is the mean of the involved pixel positions. For examples of the use of other error models, we refer the interested reader to [Rot02, Lie01].

Candidate selection & camera calibration Depending on the complexity of the scene the described clustering typically results in numbers of three up to K_{max} vanishing point candidates. From this set we exhaustively select vanishing point triples and retain only those with approximately orthogonal projective rays. Finally, from the remaining triples the one having the largest total consensus set is chosen as the final estimate of the dominant orthogonal structure.

In the case of unknown internal camera parameters, the camera calibration necessary for the orthogonality test can be carried out individually for each triple of vanishing points. For this we have chosen the composite calibration method described in [KS05], assuming square pixels and the camera's principle point to be located in the center of the image. Our experiments have shown that a further refinement of its position often caused unstable calibration results, thus we did not consider it further.

Comparison to other known methods In preliminary experiments, we compared our method to implementations of two state-of-the-art methods [KZ02a, WV07] provided by the authors. We found our algorithm to give qualitatively comparable results to the latter, however usually running five to ten times faster. Both methods performed favorably in comparison to [KZ02a], which sometimes missed vanishing points in cluttered or not Manhattan-like scenes, see Fig. 20.

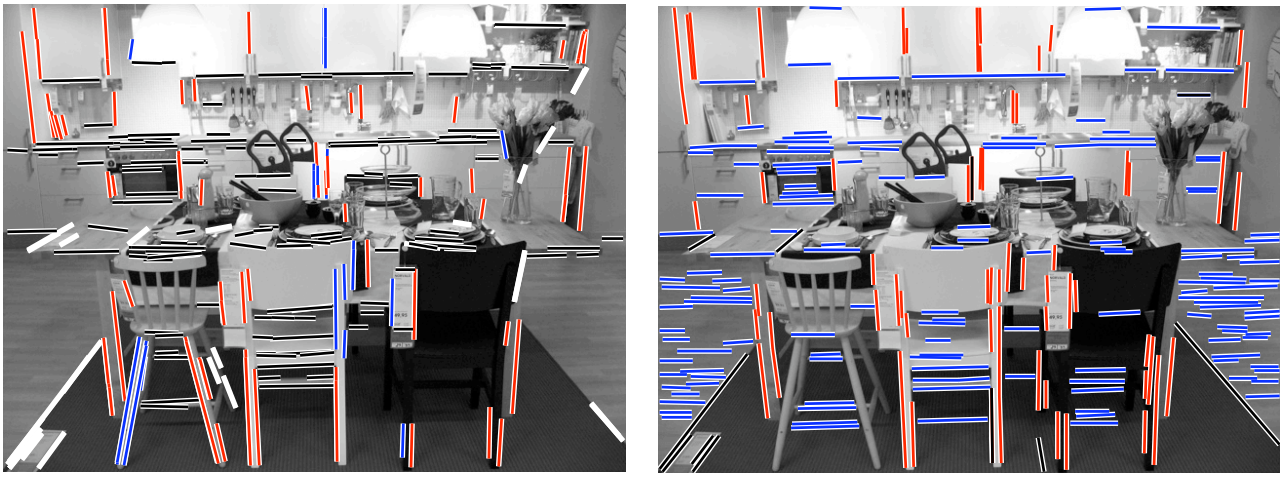


Figure 20: Comparison of the method [KZ02a] and our proposed algorithm on an image of a cluttered scene. Line sets corresponding to each of three detected vanishing points, differentiate by color, are shown. Notice that the orthogonal set of vanishing points, depicted by memberships of lines to them, was estimated incorrectly by the method [KZ02a], but correctly by our algorithm. White lines in the left image correspond to noisy lines, not associated with any vanishing point.

4.2 Monocular surface orientation estimation

In this section, we describe a novel approach devised to help a robot to understand the content of a scene, given a single image. To be more specific, we propose a method for decomposing a single monocular image, possibly stemming from an non-calibrated camera, into orthogonal planes, see Fig. 21. Finding these planes in the image can significantly aid a robot in self localization, navigation and further recognition of objects or landmarks dominating indoor environments, such as windows, doors, tables, chairs, etc..

A priori, we design a method for a non-calibrated acquisition settings to be able to also handle cases for which either the internal camera parameters are unknown, or are likely to be imprecise. In experiments it is shown that the method is able to extract a significant amount of structural information from a single monocular image. However, a later merging of entire image sequences will greatly contribute to a stabilization of the whole process.

The general concept of the proposed chain is related to previous approaches [KZ05, LMS+06, RLK05, HEH07]. However, we formulate the problem in a probabilistic graph-based framework allowing to solve it on a more global level than before. The paper is in its spirit and goals most similar to the recent state-of-the-art work of Hoiem et.al. [HEH07]. They use learnt appearance models based on various geometric, color, and texture cues to partition an image into coarse 3D surface entities. We show that even without learning and by applying less cues we can still compete with their method and often get better results.

4.2.1 Algorithmic overview

We shortly summarize the main steps leading to the final detection of orthogonal planes in a monocular image. The algorithm consists of sequential steps for the detection of

1. lines and vanishing points coming from their intersections as the largest total consensus sets corresponding to orthogonal directions.

2. quadrilaterals or their parts corresponding to rectangles or their parts in a scene.

3. orthogonal planes in a scene based on an MRF framework formulated on over-segmented image; utilizing vanishing points, ideal lines and quadrilaterals.

4.2.2 Detection of perspectively distorted rectangles

Human made environments contain many rectangular structures. These, depending on occlusions and the camera's field of view, are projected as complete quadrilaterals or incomplete parts (e.g., U- or L-shaped features) thereof. Such features represent strong visual cues for the detection of planar surfaces and consequently are of aid to the task of scene reconstruction and understanding.

In our work we use a perspective rectangle detection method related to the approach of [KZ05], however, applying a probabilistic graph-based method. To be more specific, we formulate the problem as a search for the Maximum Aposteriori Probability (MAP) solution of the MRF defined on lines consistent with the vanishing points. Such formulation allows to avoid an exhaustive search over rectangle hypotheses coming from all possible intersections of detected lines in the image. Besides its efficiency, another advantages of our approach is that it does not only detect perspectively distorted rectangles, but also sub-parts if they are compatible with the initial plane-hypothesis. However, if necessary it can be replaced with other techniques, such as the one presented in [LMS+06, HLD07]. For an example of the features found, see Fig. 21 and Fig. 48.

For the sake of simplicity, we will omit algorithmic details in the following description of the method in use. The interested reader is referred to [MWK08] for a thorough presentation of the technique.

MRF-based rectangle detection The main steps of our method are the following.

1. Line segments and vanishing points are localized and if necessary used for camera autocalibration.

2. Each line segment is assigned to its corresponding vanishing direction and line segments compatible with a vanishing line, i.e., the two vanishing points generating it, are grouped by principles of proximity and continuity.

3. A graph representing the MRF is constructed from the detected line segments respecting vanishing direction assignment and geometric properties between pairs of the neighbouring lines; encoded via data and smoothness terms.

4. The MAP is computed yielding a unique label, representing one of four rectangle edges, assigned to each line segment such that meaningful rectangles are established.

4.2.3 MRF based plane detection

Having detected vanishing points and lines pointing to them we want to assign to each pixel in an image its 3D orientation w.r.t. to a camera coordinate system. As we assume a Manhattan world structure, this is equivalent to assign one of three labels, where each label corresponds to one of three orthogonal planes, to each pixel.

To solve the problem on a global level, i.e. to allow to take into account prior information about possible pixel orientations and relations between neighboring pixels simultaneously,

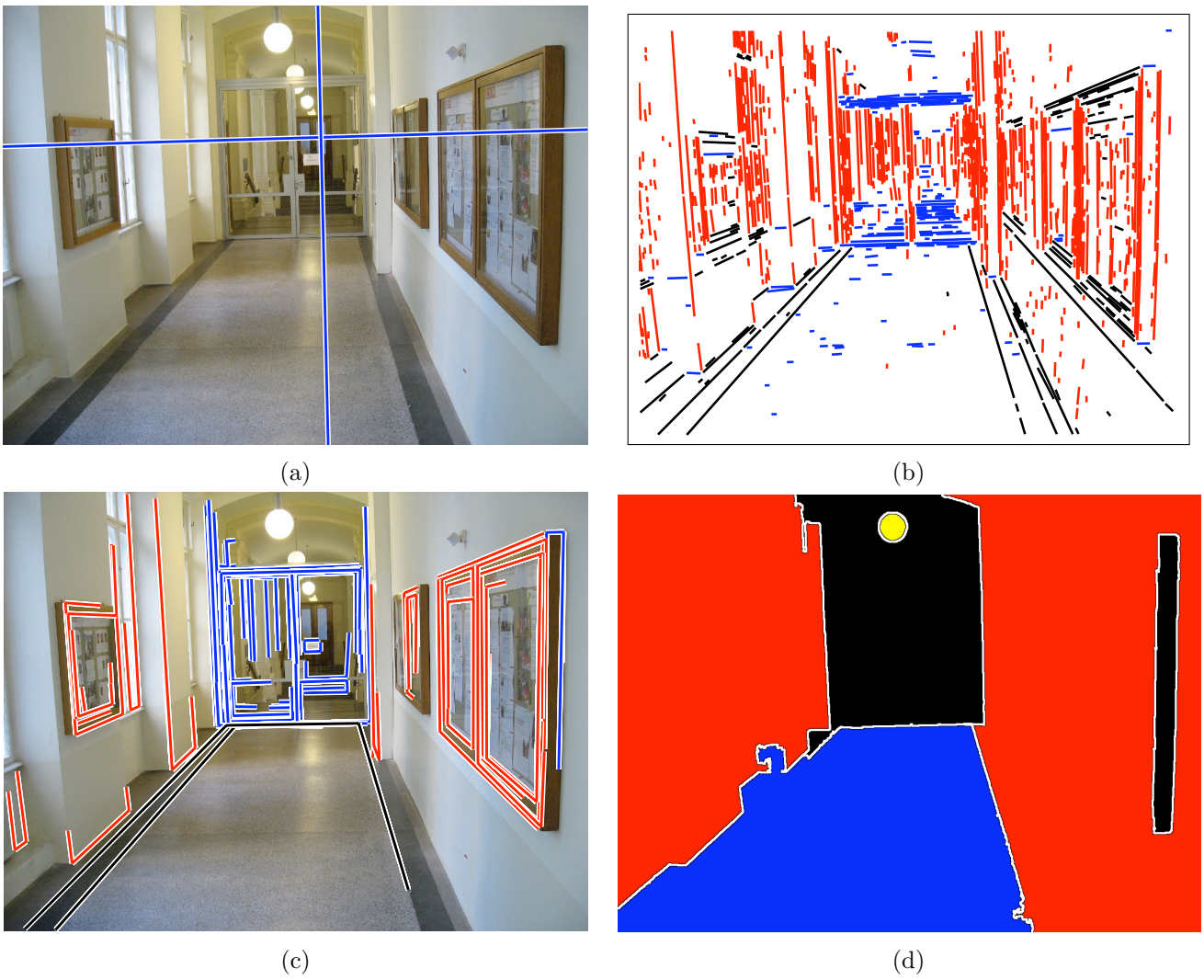


Figure 21: Proposed sequential chain leading to detection of orthogonal planes in a monocular image. (a) The input image (844×1126 pixels) with vanishing lines depicted. (b) Detected lines consistent with three automatically estimated orthogonal vanishing points. (c) Detected partial and complete quadrilaterals utilizing the vanishing points and lines pointing to them. (d) Final segmentation of planes based on a Markov Random Field formulation employing vanishing points, lines, and quadrilateral segments.

we formulate the problem in a fully probabilistic framework; as searching for a maximum posterior (MAP) configuration of the Markov Random Field (MRF) [YFW05]. It has been shown [Wer07] that the solution can be found as a Gibbs distribution with maximal probability, i.e., by solving the so called labeling or Max-sum problem of second order - maximizing a sum of bivariate functions of discrete variables.

We assume an MRF, i.e., a graph $\mathcal{G} = \langle \mathcal{T}, \mathcal{E} \rangle$, consisting of a discrete set $\mathcal{T}$ of objects (in the literature also called sites, or locations) and a set $\mathcal{E} \subseteq \binom{|\mathcal{T}|}{2}$ of pairs of those objects. Each object $t \in \mathcal{T}$ is assigned a label $x_t \in \mathcal{X}$ where $\mathcal{X}$ is a discrete set. A *labeling* is a mapping that assigns a single label to each object, represented by a $|\mathcal{T}|$-tuple $\mathbf{x} \in \mathcal{X}^{|\mathcal{T}|}$ with components x_t.

An instance of the Max-sum problem is denoted by the triplet $(\mathcal{G}, \mathcal{X}, \mathbf{g})$, where the elements

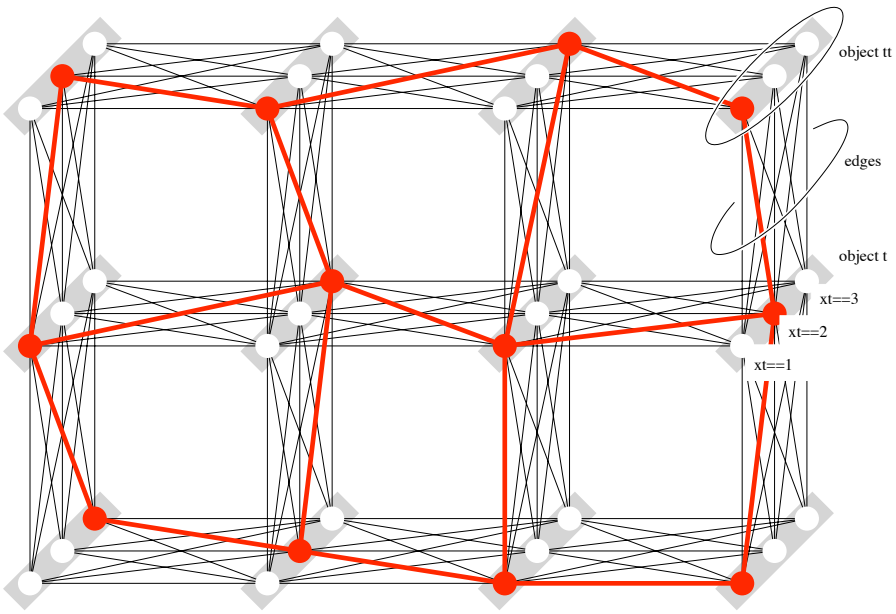


Figure 22: An example 3×4 grid graph $\mathcal{G}$ for $|\mathcal{X}| = 3$ labels with symbols explained in the text. A labeling $\mathcal{L}$, i.e. solution, from Eq. (2) is shown by a red thick subgraph. Image provided by courtesy of T. Werner [Wer07].

$g_t(x_t)$ and $g_{tt'}(x_t, x_{t'})$ of $\mathbf{g}$ are called *qualities*. The quality of a labeling $\mathbf{x}$ is defined as

$$F(\mathbf{x} \,|\, \mathbf{g}) = \sum_t g_t(x_t) + \sum_{\{t,t'\}} g_{tt'}(x_t, x_{t'}). \tag{1}$$

Solving the Max-sum problem means finding the set of optimal labellings

$$\mathcal{L}_{\mathcal{G},\mathcal{X}}(\mathbf{g}) = \operatorname*{argmax}_{\mathbf{x} \in \mathcal{X}^{|\mathcal{T}|}} F(\mathbf{x} \,|\, \mathbf{g}). \tag{2}$$

Graph entities Generally, the most difficult problem and art connected to MRF based methods is to encode all possible priors about objects being labeled (e.g., orientation, texture, color, shape, appearance) into a graph, i.e., a MRF, while still keeping the problem tractable. The priors we utilized lead to partitioning an image into geometrically and color coherent regions as Fig. 21 shows.

We build a graph on an over-segmented image, i.e., on superpixels, see Fig. 23, to keep the running time in reasonable bounds. The idea is to locally merge pixels with similar color together. The use of superpixels significantly reduces the number of objects in the graph, still preserving texture information. Simply reducing the image size and building an MRF on pixels to avoid the large complexity as implemented in many approaches leads to losing details and high texture frequencies. In this paper, we use the fast Minimum Spanning Tree based method by Felzenszwalb [FH04], giving us, by appropriate setting of parameters, 500-800 regions on average. However, any other over-segmentation can be used.

The graph entities are the following. The superpixels represent objects, i.e. the set $\mathcal{T}$, in the graph and edges, i.e. the set $\mathcal{E}$, are established between each two neighboring superpixels. The number of nodes (labels) K is 4, i.e., we use one label for each orthogonal plane and one label for "undecided" to allow the solver mark the places where there is not enough information to decide which plane the superpixel belongs to.

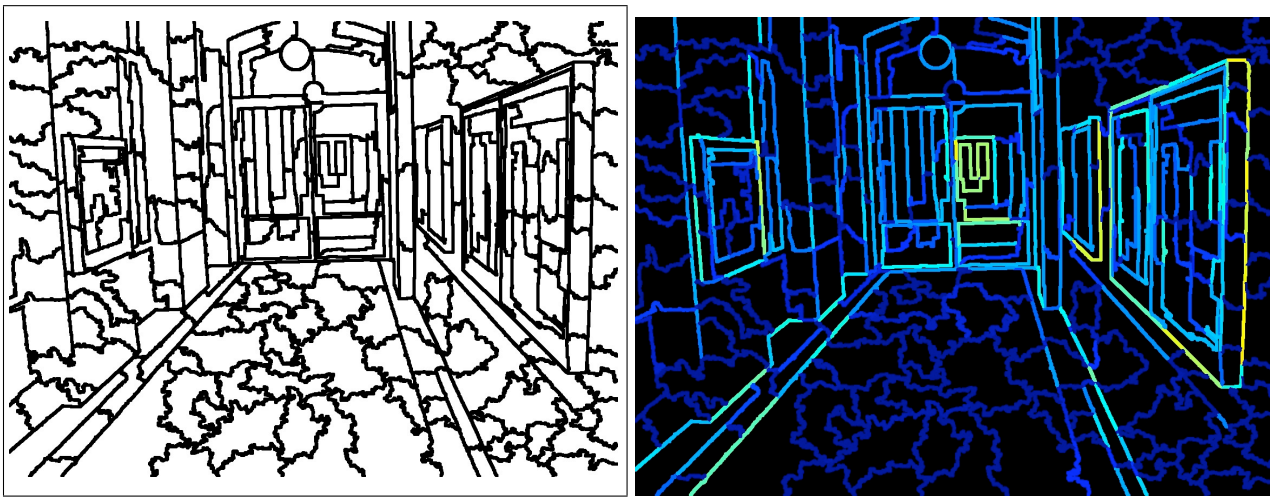


Figure 23: Left: Superpixels detected in the image from Fig. 21. Each region corresponds to one object in the constructed graph. Right: The smoothness term. Boundary-color encodes the penalty set in the graph between the objects corresponding to two neighboring superpixels. Darker coloring denotes less penalization. Note, that straight boundary segments are penalized stronger.

Each edge $g_{tt'}(x_t, x_{t'})$ and each object node $g_t(x_t)$ is set accordingly to the smoothness and data term respectively, described in the following sections. After building and setting the graph, the Max-sum solver [Wer07] is run to obtain a particular label x_t for each superpixel t.

Smoothness term The smoothness term, defined by $g_{tt'}(x_t, x_{t'})$, controls the mutual bond of neighboring superpixels. In our case we take into account a color difference between superpixels and a straightness of the common boundary. It can be written as follows

$$g_{tt'}(x_t, x_{t'}) = \exp\left(\alpha \|\mathbf{u}_t - \mathbf{u}_{t'}\|^2\right) - \beta\, S^{\mathrm{st}}_{tt'}, \tag{3}$$

where $\mathbf{u}_t$ is a 3-element color vector of the t-th superpixel (mean color of all pixels belonging to that superpixel) and $\alpha < 0$ is a parameter pre-set to -10. We represent $\mathbf{u}_t$ in the Lab color space because of the perceptual non-uniformity of the standard RGB space. $S^{\mathrm{st}}_{tt'} = \frac{\sum_i^N \mathrm{length\_line}_i}{\mathrm{length\_boundary}}$ is a sum of lengths of N lines fitted to the shared boundary between two superpixels t and t' (longer than 20 pixels), see Sec. **??**, normalized by the length of the boundary. The parameter β controlling the influence of the smoothness term, was set to 0.5 in our experiments.

The proposed smoothness term in Eq. (3) penalizes connections between superpixels with similar color and jagged boundaries less, thus tends to merge them. Such jagged boundaries are usually produced accidentally due to weak gradients [FH04] and therefor do not correspond to real splits of two superpixel patches in the scene. Thus, it is preferable to force their merging.

Data term The data term $g_t(x_t)$ encodes the quality of assigning a label x from the set $\mathcal{X}$ to an object/superpixel t in the graph. The quality measures how the superpixel itself suits to particular class models, in our case, to lie on one of the orthogonal planes.

For each superpixel we need to set 4 numbers, i.e., how likely is that the superpixel is marked by one of four labels. The first three labels stand for the belief that a superpixel lies on one of the three orthogonal planes; the forth label encodes the level of "undecidedness".

The consistency of a superpixel to a plane is expressed via a deviation of gradient orientations of the pixels along the boundary of the superpixel to two vanishing points corresponding to that plane. For computation of the gradient orientations we use the 5-component gradient mixture model described in [CY03]. For each image pixel, the model gives the probability of the pixel lying on an edge, the membership to one of the three vanishing points, and the probability of being noise, i.e., not being compatible with any vanishing point. We take into account only those pixels having a probability of being on an edge above a certain threshold. Then, a normalized histogram $h_t(y)$ with four bins $y = \{1, 2, 3, 4\}$ is computed from vanishing point memberships of all pixels lying along the t-th superpixel boundary. The fourth bin accumulates the aforementioned noise term. For an example, see Fig. 24.

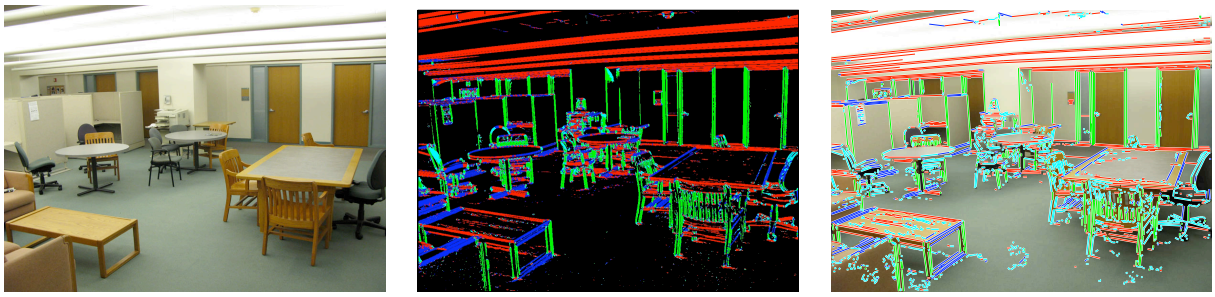


Figure 24: Example of the five-component gradient mixture model. Left: Original image, Middle: Gradient mixture model: red, green, and blue denote gradient orientations compatible with the vanishing points, cyan denotes noise, black off-edge gradients, Right: Line segments compatible with vanishing points.

Finally, the consistency of the superpixel with each label is set as

$$g_t(x) = \begin{cases} \sum_{\substack{i=1 \\ i \neq x}}^{3} h_t(i) & \text{if } x = \{1, 2, 3\}, \\ h_t(x) & \text{if } x = 4. \end{cases} \tag{4}$$

In the data term, two additional priors are utilized. One stemming from the position of ideal lines and one from detected quadrilateral segments. The ideal line is defined as a line passing through two vanishing points and is a projection of an intersection of a 3D plane with a plane at infinity [HZ04]. It gives us the constraint that a superpixel detected in the image cannot cross the ideal line of the plane it belongs to. The data terms of such superpixels are set to zero to decrease the belief of them to lie on a particular plane. Fig. 21 shows two ideal lines where one corresponds to a ground plane. Notice that this line, called a horizon, is completely above the ground plane and therefore superpixels on that plane cannot cross the horizon.

The second prior comes from the fact that all superpixels behind detected quadrilateral segments, see Sec. 4.2.2, have to lie on the plane where the segments are detected. The data terms of such superpixels are increased or set to a high value in order to strengthen the belief of them to lie on that particular plane.

4.2.4 Results

We evaluate the proposed method on large variety of indoor images downloaded from the Internet. Some of the most representative are shown in Fig. 25. The images are approximately

1 Mpixel large and their quality varies since they were taken by different, to us unknown, cameras under different illumination conditions. The results show feasible and stable performance, although light reflections, shadows, jpg-artifacts, and occlusions, are present in the images.

Fig. 25 shows each image segmented into 4 labels, three for each orthogonal plane and one for "undecided". We compare our method to the state-of-the-art method [HEH07] aiming at exactly the same goal, i.e. at recovering surface layout from a single image. To produce the results of [HEH07] the publicly available code[5] was used in combination with a provided indoor classifier. The presented results show comparable performance of our method, often achieving better result. Moreover, the run-time of our method was shorter, 1 min on average, while the method of [HEH07] took 3 min using the same Pentium 4@2.8 GHz.

The proposed method is currently mostly implemented in unoptimized MATLAB and many of the routines and functions can be re-implemented in much more efficient way in C/C++. Our experience and preliminary results indicate that the running time could be decreased to ∼5 to 10 s. For finding the MAP of the MRF we use a publicly available[6] C++ implementation of the Max-sum solver [Wer07].

It can be seen in Fig. 25 that at some places, especially at connections of planes, our result is not always correct. This is caused by either superpixels missing the true boundary and thus overlapping two planes. Or, there is an occlusion present, i.e., one plane partially occludes the other. In the second case, the incorrect behavior comes from the data term formulation, Eq. (4), as the superpixel is expected to contain two strong gradient directions only. In the case of the occlusion, e.g. a table leg touching a floor, the superpixel covering a part of the floor and touching the leg contains pixels at its boundary which are pointing to a vertical vanishing point. This may cause that the superpixel is incorrectly assigned to one of the vertical planes. The resulting inconsistency, depending on neighboring superpixels, cannot always be solved by the smoothness term.

4.3 Scene planes from lines

One of the features of a scene useful for furniture recognition may be planes. In this section, the detection and extraction of such scene planes is presented.

4.3.1 Stereo vision

The 3D vision problem is illustrated in figure 26. The physical point M is imaged in the two retinal planes as m_1 and m_2. The distance between the two camera centers C_1 and C_2 is called the baseline. The intersection of the baseline with the two retinal planes defines the two epipoles e_1 and e_2. The so called epipolar geometry – which is usually considered for the search of corresponding points – is formed by the geometric relation of the image points m_1 and m_2. As shown in 26, the 3D point M, the two image points m_1 and m_2, and the two camera centers C_1 and C_2 are coplanar, i.e. they lie on the same plane, denoted as π. This plane together with the baseline as axis forms a pencil of planes. The most interesting property of this construction is that the ray in 3-space defined by by the camera center C_1 and 3-space point M is imaged as the line l_2 in the second view. As the 3-space point M must lie on this ray, the image of M has to lie on l_2 [HZ03].

In stereo vision, there are two main problems to be solved [Fau93]. The first one is called the *correspondence problem*: For a point m_1 in image one, decide which point m_2 in image

[5] http://www.cs.cmu.edu/~dhoiem/projects/software.html
[6] http://cmp.felk.cvut.cz/cmp/software/maxsum/

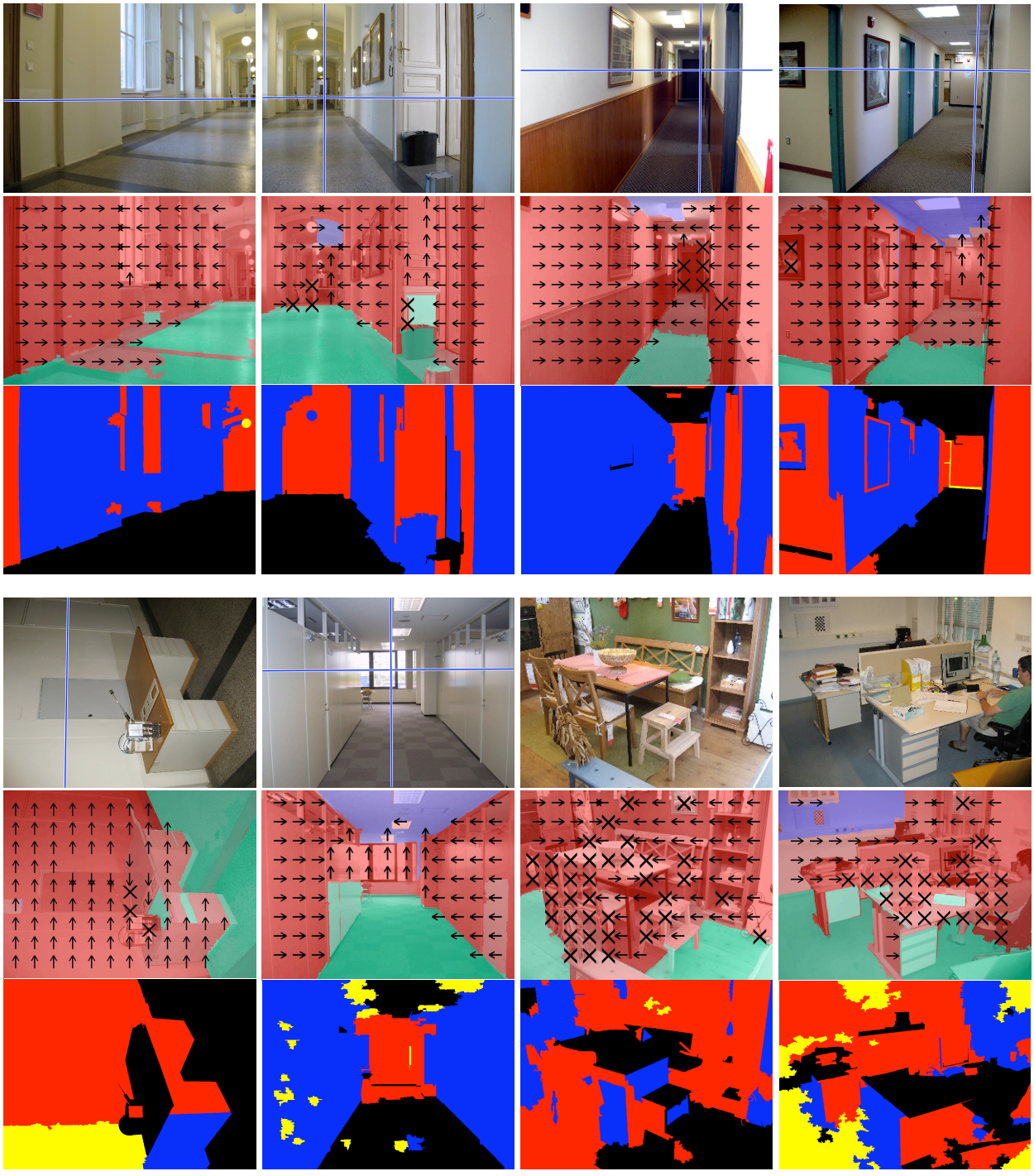


Figure 25: Results of planes-detection in single indoor images arranged in triplets. Top: Input image with in-plotted ideal lines estimated by our method. Middle: The method by Hoiem et.al. [HEH07] segmenting images into ground plane and vertical planes. Arrows stand for plane orientations to the left/up/right, markers 'o' and 'x' for porous and solid materials, respectively. Bottom: Our proposed method segmenting images into three orthogonal planes. Each plane is depicted by a different color, where yellow color stands for "undecided" pixels.

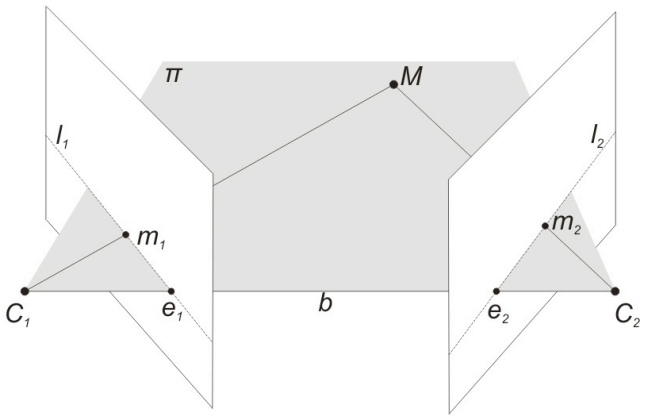


Figure 26: The 3D vision problem.

two corresponds to the point in image one, i.e. which point m_2 images the same 3-space point M as m_1. The second one is the logical sequel to the *corresponding problem*, called the *reconstruction problem*: For a given pair of corresponding image points m_1 and m_2, compute the 3D coordinates of the 3-space point M. Theoretically, this problem can easily be solved by intersecting the rays $\langle m_1, C_1 \rangle$ and $\langle m_2, C_2 \rangle$. However, in practise, the image points are not perfectly known and therefore the rays may not intersect. Additionally, the result of the reconstruction heavily depends on the accurate knowledge of the image planes and the camera centers in the world coordinate frames, which is determined through calibration.

A stereo correspondence algorithm, which works with points or point-like features can benefit from the constraint that the corresponding point m_2 will lie on the associated epipolar line and therefore the search space can be restricted to search along the line l_2.

Using features for matching other than points like in our case lines, this strong constraint does not hold.

4.3.2 Lines and line matching across two views

Line segments have some useful properties compared to points for tasks like scene reconstruction, especially when it has to be done fast, to be feasible for mobile robotics. Line segments are much more discriminative than points and particularly for indoor scenes, they are the features of choice for describing the planar scene parts. However, these useful properties don't come for free. Line matching across views is still a challenging problem. Difficulties include:

- The orientation of a line segment can be extracted very accurately but the endpoints are not reliable [SSZ97]. Furthermore, the error induced by the endpoints varies with the length of the line segment, which doesn't make the overall process simpler.

- The topological connectivity of the line segments is often lost during line extraction stage. Line segments in two views of a shared edge in 3-space are often broken in different amount of segments in each view, inducing matching problems in later stages if not corrected.

- At the matching stage: No strong disambiguating geometric constraint available in a stereo setup. For lines with finite length, the epipolar constraint can be be applied to the endpoints, resulting in a weak overlap constraint. As we will show later on, this constraint does not hold for our environment. For infinite lines there is no geometric constraint at all.

- Line segments have little distinctive appearance and therefore the attributes for describing a line are weak: The length of a line can vary strongly as the extend of overlap. The orientation is the most stable attribute, but almost almost useless without other additional features. Using the intensity neighborhood of the line with correlation techniques for pointwise matching of line segments needs additional precautions [SSZ97]. Point to point mapping has to be established with the epipolar geometry and the correlation patches have to be corrected.

- Matching groups of line segments leads to more geometric and topological constraints and can be solved by graph matching, but the disadvantage is the increased complexity and therefore the increased computational costs.

Line segment correspondence In Figure 27, the extracted line segments are overlaid in the images. The problem is clearly visible – there are several matching cases:

- (2) No correspondence: A line segment in one image has no corresponding line segment in the second view.

- (1) Full correspondence: Two line segments correspond and they overlap or partially overlap.

- (3) Multiple correspondence: Several line segments correspond and overlap the line segment.

- (4) Virtual correspondence: This line segment corresponds to line segment marked with (1), but there is no overlap of the segments.

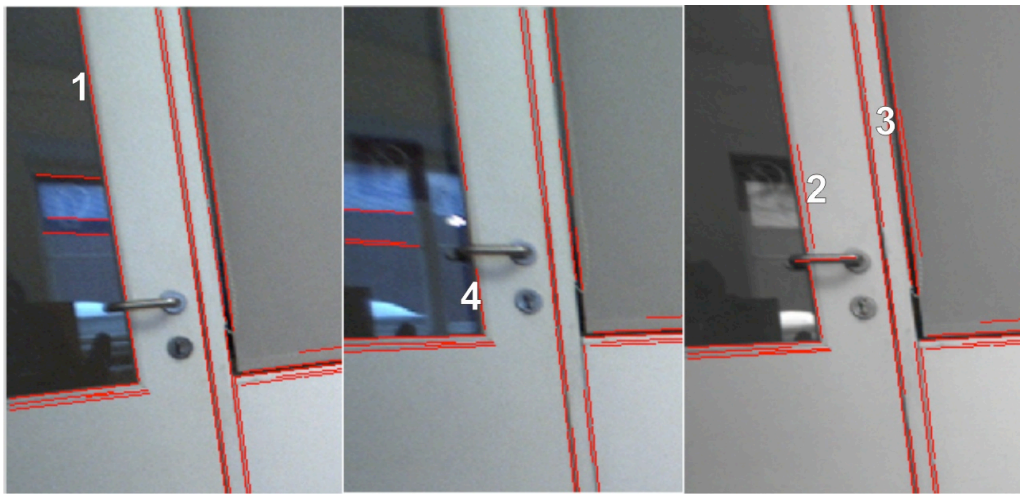


Figure 27: Finding the correct correspondence is no trivial task.

Color In indoor environments, most parts of the scenes are poorly textured or even untextured. One method of extracting discriminative features out of line segments was proposed in [BBFVG05]. They extracted color profiles at each side of the line segment and used a partitioned HSV color space for illumination invariant matching. In addition to the color profiles, they used a topological filter to increase the matching score. Inspired by their work, we

started building an line segment color profile matcher. As our goal is fast running algorithm, we discarded the idea of matching color profiles as 166×166 matrices with HSV color space, as they are too computational expensive. Instead of that, we use cummulative histograms in the L*a*b color space together with an angular constraint arising from small baseline stereo for matching.

Epipolar geometry Trivially following from the nature of epipolar geometry, horizontal stereo has problems with horizontal aligned features and vertical stereo with vertically aligned features. As the robot will mostly drive aligned with the room layout (lots of lines are imaged horizontal in the image), i.e. along floridors, along walls, we decided not to use a standard stereo setup with epipolar lines aligned horizontal. We mounted the cameras aslant, so the epipolar line crosses the image at an angle of about 45 degrees 28. This is a helpful construction for the reduction of the possible line matches and it discards lots of possible mismatches. As in man made environments, most of the structures are vertical or horizontal, the best compromise was to mount the cameras aslant.

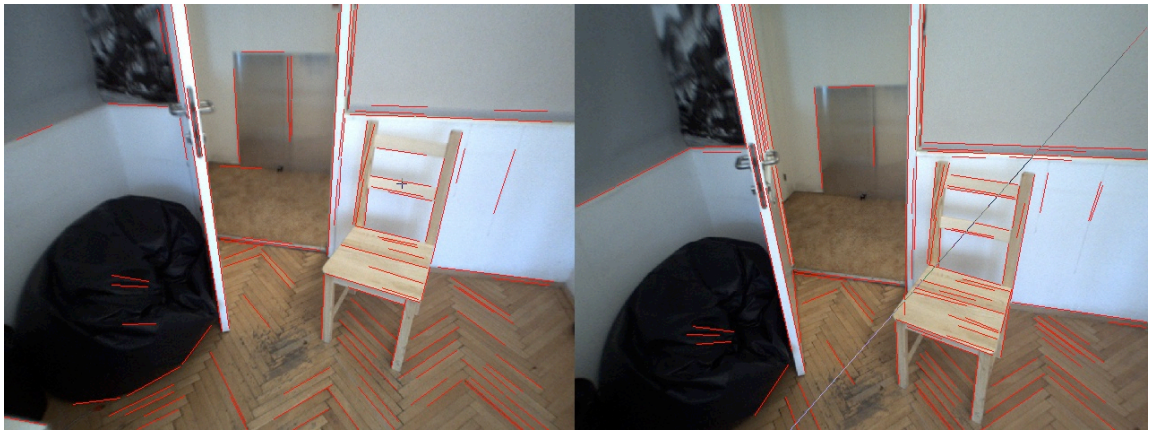


Figure 28: Extracted line segments (red) in two vies and the corresponding epipolar line for a point on the chair in left image.

Repetitive structures One of the biggest problems in matching lines across views in a stereo setup can be seen in Figure 28. The repetitive structure on the ground floor leads to a large amount of mismatches. First of all, small changes in the viewing direction together with the changed illumination conditions causes the straight line extraction stage to not extract the same line segments in both images. Furthermore, the line segments are well aligned among each other and have mainly the same color properties. As they look quite the same and are aligned like the same, lots of mismatches cannot be detected and therefore not corrected.

Line reconstruction Triangulation of lines is easier to accomplish, as two arbitrary planes generally intersect in 3-space. Lets take a closer look on the triangulation of lines. A line segment in an image and the associated camera center induce a plane in three-space, see Figure 29. As the corresponding line segment in the second view also induce a plane, the intersection of these two planes is the line L in 3-space. Like straight lines always intersect in projective 2-space, planes are their equivalent in 3-space. As intersection behind the two cameras are meaningless, they can easily be detected as wrong line matches and can therefore

easily be removed. For 3-space-lines, there is no method for distinguish between a correct matching of line segments and a wrong matching.

Assuming that all line matches are correct, there is still a problem of triangulating lines: Unlike point which have to fulfill the necessary property of being on the corresponding epipolar line $x_1 F' x_2 = 0$, lines don't have such a property and moreover they have no algorithm like point features to enhance their accuracy. Two lines with their imprecisions are triangulated as illustrated in 29 and their error is not just propagated, the error is increased. Furthermore, the resulting 3-space-line has no length. Yes, you can take the outmost points projected by the two line segments, but the resulting endpoint of this line have no meaning in the real world. Therefore, the simple and straightforward idea of triangulating lines and than working further in 3-space leads to no satisfying methods and results.

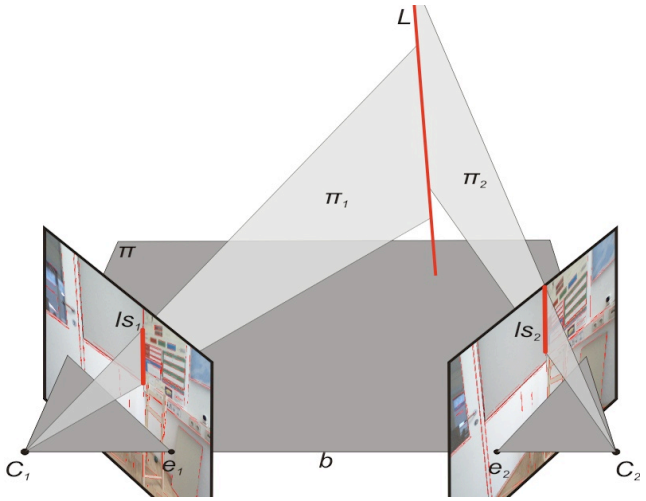


Figure 29: Triangulation of a corresponding line segment pair.

Findings on lines and stereo Lines are a rich source of information about our environment. They are much more meaningful than points and point features and they are the right choice especially in man made indoor environments. Compared to point or area based descriptors like Surf, SIFT, MSER, etc. lines are much faster to extract. And here the advantages end and problems begin to arise. Long lines are more discriminative than short lines, but they are much harder to extract. Lines often break apart in the extraction phase and there is no standard algorithm which works fast and robust to merge these small line segments. Line descriptors are not that distinctive as needed. At the matching stage, mismatches are on the order of the day and cannot be detected nor corrected. Triangulation of the line segments leads to inaccurate lines in 3-space as two line segments place no supplemental constraint in the triangulation process and therefore cannot be optimized for accuracy as points can. Despite all problems, lines are the feature of choice and in the next section, a way out of the problems is presented.

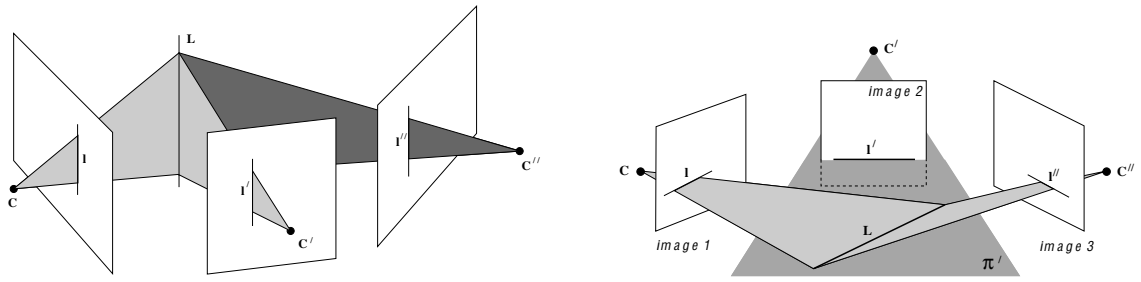
4.3.3 Trinocular stereo vision

With today's decreasing costs of imaging hardware, money is no longer a factor against three cameras. Commercial products like point grey's digiclops[7] are available and Matlab toolboxes

[7] http://www.ptgrey.com/products/digiclops

for calibrating multi-camera systems are also freely available [SMP05]. Using three views is far from being standard as it is stereo nowadays. But the arising geometric constraints on lines from using three cameras are the ones we need for our task: Fast and reliable matching of lines and accurate triangulation of these.

In a stereo setup, see Figure 29, the two planes in projective 3-space always intersect. Having three cameras, the specific characteristic that three planes intersect in one single line - Figure 30(a) places the needed constraint on the triangulation process. Line matches can be evaluated and line triangulation can be optimized.



(a) Trinocular stereo: The geometry of three cameras observing the same 3-space line. [HZ03]

(b) Line transfer from one view to another via a plane in the third view. [HZ03]

Figure 30: Three view geometry.

The trifocal tensor and matching of lines The trifocal tensor - the 3-camera-equivalent to the fundamental matrix in two views - provides a useful property for line matching, called *line transfer via the trifocal tensor*. Suppose three corresponding lines in three views Fig. 30(b). The geometric relationship of the setup inherent in the trifocal tensor allows one to transfer the line l from image 1 via the plane π - induced by line l and the associated camera center C - to image three. This property can be used successfully for verification of line matches. This is the dulcet theory. In practise, this verification via the trifocal tensor depends heavily on the accuracy of the trifocal tensor. Furthermore, there are configurations where wrongly matched line segments cannot be detected, but this is a rare situation.

4.3.4 Scene planes

As we are searching for basic structural entities for describing the world, lines are appropriate for further grouping to planes. As simple bottom up grouping leads to erroneous structural basic features, a more elaborate method which utilizes the projective geometry is needed. The method of choice is plane sweeping with homographies. Scene planes induced by line segments can be seen in Figure 31; A line correspondence creates a pencil of planes in projective 3-space. All possible homograpies having this line as axis, can be parameterized by a single projective parameter μ. Figure 32 illustrates some steps in the homography sweeping process.

Plane hypothesis generation This method seems to be a good basis, as it was used several times in the past. The homography can be calculated directly from one line and one point correspondence and the epipoles[HZ03]. As we have no point features in our scenes - virtual lineintersection could be used - and line segments often correspond to two planes, a sweeping approach is justifyable.

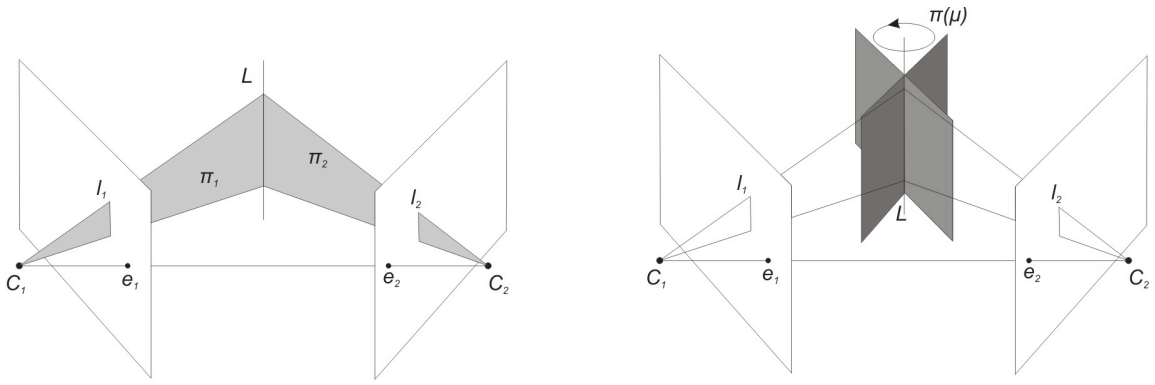


Figure 31: A line correspondence induces a pencil of planes.

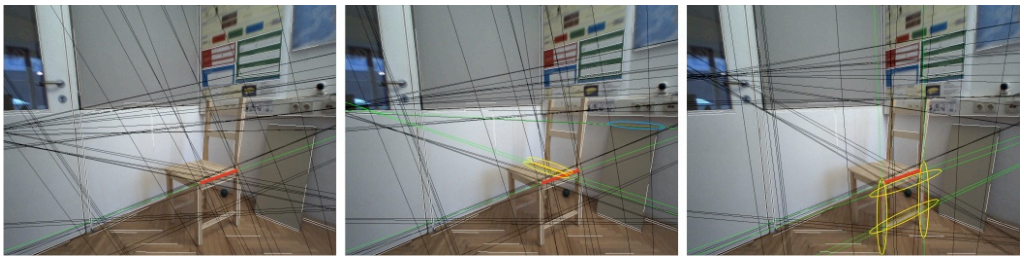


Figure 32: Lines are transferred across the images via the homography. At each rotation step, the line matches are calculated.

Figure 33 shows some of the extracted planes of the scene. As is is also clearly visible in the illustration, there are some line correspondences to scene planes - Figure33 middle image - which do not correspond to the *real* object. Therefore, these planes are nothing more than hypotheses: We must verify them.

Plane hypothesis verification In addition to some small errors in plane hypotheses generation which could be corrected by additional robust fitting of the homographies using Ransac or LMedS, there are some hypothese which are clearly incorrect, i.e. they represent no *real* scene plane. This errors can be corrected by using all three views for verification and additional techniques. For example, as scene planes are 2-dimensional, line segments corresponding to the vanishing points of the plane can reduce the mismatches. Furthermore, warping image patches and calculating simple difference - this is possible as the homographies eliminate the projective interview-distortion - is possible. These additional techniques will make it feasible to eliminate scenarios like Figure 34.

Conclusion on scene planes The trinocular setup with two color cameras and one monochrome camera made it feasible to create a fast, stable and dependable process of obtaining basic structural entities or parts.

4.4 Dense stereo-based pre-segmentation of pieces of furniture

4.4.1 Ground Plane Estimation

Ground plane estimation uses the fair assumption that an estimated camera orientation towards the ground plane is known from the robot set-up and the head and robot kinematics

Figure 33: Scene planes in transparent red: left picture illustrates the scene plane of the wall, middle picture a scene plane of a chair and right image shows a horizontal scene plane of the chair. cH is the convex Hull of the four lines corresponding to the scene plane π

.

Figure 34: A scene plane hypotheses: due to inaccuracies in line extraction and trifocal tensor estimation, some lines are wrongly classified as corresponding lines to a plane. This hypotheses has to be rejected, as it clearly is no scene plane.

or from the previous tracking step [Bur02]. Even when going over floor discontinuities this estimate is sufficient to render fast ground plane estimation efficient. Throughout the complete approach this assumption is exploited but with very loose constraints to make sure that all potential good data points even if very noisy are retained. Consequently large uncertainty values will be used. Robustness is achieved by a sequence of filtering operations all using loose constraints. The ground plane is then found with a probabilistic scheme to rapidly detect the most likely plane in three steps.

First, three points are randomly sampled from the image data. These points are sampled from the lower third of the image, because navigation and viewing angle indicate in this region the highest likelihood to find ground plane data, however a pre-sampling as in [Bur02] has not been found necessary. A second criteria is to estimate the height of each point using the estimated camera orientation and the point disparity. The distance d to each point in the scene can be estimated because a rough pose of the camera is known. Then the sample points for the ground plane fit are only used if the estimated point height $\hat{h}$ (see Fig. 35) lies within 30% of the expected height h, where

$$\hat{h} = d * cos(\theta) \tag{5}$$

is calculated from the viewing angle θ.

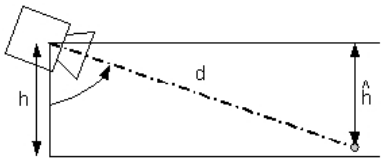


Figure 35: Estimating the height $\hat{h}$ of a data point. The viewing angle is θ.

Second, the normal vector of the plane is calculated from these three points. If the normal vector deviates not more than, e.g., 10 degrees from the expected ground plane, the surface is retained as a potential hypothesis.

Steps 1 and 2 are iterated until 20 plane hypothesis are found. Each hypothesis is then weighted by taking a random sample of 500 data points. This renders the process very efficient, since taking all data points would be too time consuming. For each of the 500 points the distance to the plane hypothesis is calculated and the median (originally the mean [Bur02]) is calculated. To the median-filtered points a plane is fit by calculating the Eigenvectors of the covariance matrix. The cross product of the Eigenvectors corresponding to the two larger eigenvalues gives the normal to the estimated plane. It is again checked if the plane lies within the natural bounds of tracking and the predicted camera pose (i.e., a deviation of less than 10 degrees is used). We adopted the median as measure of quality since this enables to better reject outliers.

After eliminating the data points of the ground plane and individual outliers a disparity image such as in Fig. 36 is retained to segment the table.

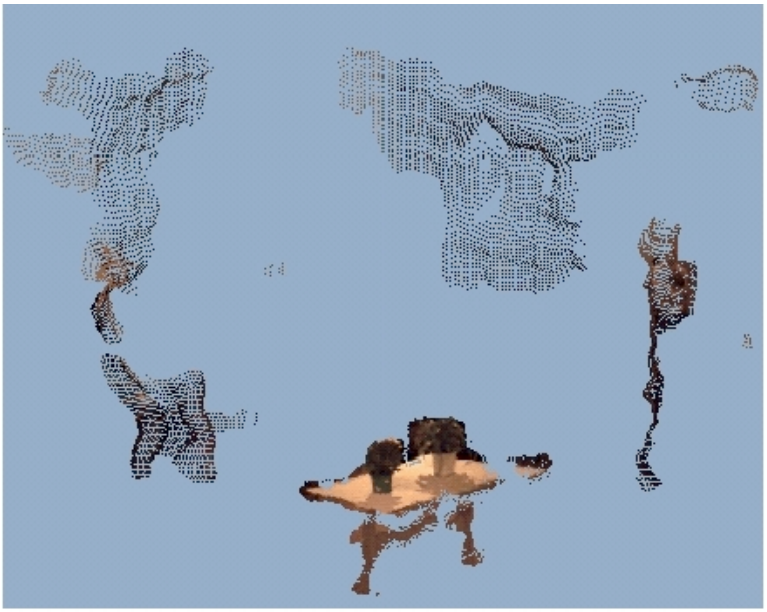


Figure 36: Data points after ground plane and outlier elimination.

4.4.2 Table Detection and Table Height Estimation

With the fit of the ground plane, only the remaining data points need to be considered to locate tables (Fig. 37). A few spurious data points are eliminated using the obvious constraint that tables are at least $10cm$ above the floor and not higher than $150cm$. The task at hand is to efficiently group pixels into horizontal planes. The task is different to locating the ground plane, which can be assumed to extend over a larger area. Tables might be small and only narrow regions, since the disparity image might contain only data of the table rim and little or no data of a textureless table surface. Due to the restricted space the approach is described as best as possible and exemplified with details about one example.

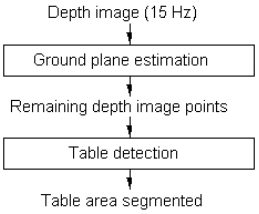


Figure 37: Flow diagram of the approach to locate tables in 3D.

A direct probabilistic search for tables, as it was done for the ground plane, is not feasible. The reason is that tables are much smaller regions and random sampling would require too many samples. Hence, the approach exploits the image-based neighbourhood of the original 2D image data to locate compact regions in 3D. To also find small tables the assumption is that at least the table rim of several pixel width and length has been detected in the disparity image. Because the table rim is a discontinuity, this is a not restrictive assumption, which the subsequent experiments will confirm.

The data points after ground plane detection (see Fig. 36) are investigated. The points are ranked depending on the pixel density in the immediate neighbourhood and either marked as potential table cluster or disregarded. For each point retained this list, $P1(x, y)$, two neighbours $P2$ and $P3$ are found in the interval $P2[x+\pm 2-\pm 5, y\pm 4]$ and $P3[x\pm 4, y+\pm 2-\pm 5]$. For any triplet $P1, P2, P3$, a plane is uniquely determined and if the normal vector does not deviate more than a threshold, the experiments will all use 20 degrees, the triplet is a valid table hypothesis. The result is a weighted list of table hypotheses.

In the next step the table hypotheses are tested against the remaining data points. Again the method using the median is utilised (see Section 4.4.1). To locate a table the neighbourhood relationships of the 2D image are again exploited. While clustering data points in 3D is cumbersome, connected component analysis of the 2D points corresponding to the data that fits to a plane is highly efficient. It can use standard procedures [Gon02] and enables to detect tables of the same height but at different locations. For an example see Fig. 38, which gives a clear peak in the histogram of the table height. The height is specified in relation to the estimated ground plane. If two tables are directly co-located, this situation would be treated as one table, a fair assumption without further cognitive perceptual processing.

This clustering procedure is executed for all table hypothesis. The result is a list of tables and the corresponding data points. To each cluster of points a plane is fit using the method given above by calculating the Eigenvectors. Fig. 39 gives an example of a table detection.

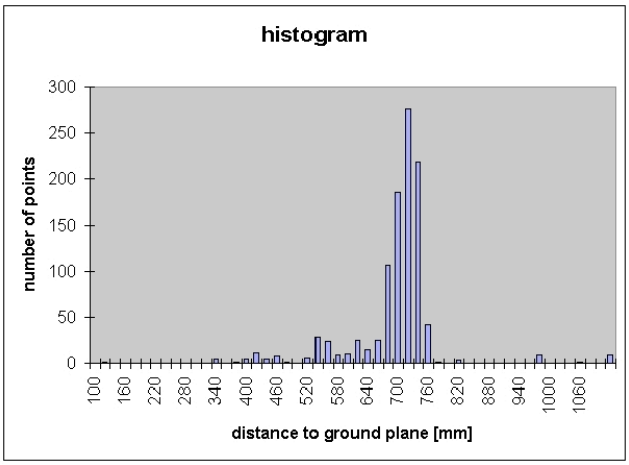


Figure 38: Histogram of table height estimation. The table was manually measured to be 72 cm high. The table height estimate gave a result of 72.4 cm.

The circumference of the table is difficult to estimate, since the objects on the table occlude larger portions of the back of the tables. A method tested is to project all points clustered to belong to the table into the table plane and to subscribe a rectangle whose sides are parallel to the first two Eigenvectors and whose size is taken from the extension of the data points. However, due to the occlusions this process is rather inaccurate, although for this example it gave best results and the table size of 1100×680 was estimated as 1194×712 with the axis slightly rotated to the right in Fig. 39. Overestimating the size is due to this rotation of the main axis and due to the smoothing of the pixels when obtaining the disparity image. While the results for this example is satisfactory, for the experiments below sometimes only a triangular part is visible and in this case the estimation process cannot be used. It will be further work to approach the table and to fuse several views to more accurately estimate the table size.

4.4.3 Experiments

The goal of the Experiments is to demonstrate that table finding is robust under typical variations encountered in an in-door home environment. There are no restrictions on the relative orientation between robot and tables. Tables can be of normal height, small couch tables or even the horizontal surface of chairs. The only assumption is that a table is at least 100 mm above the floor, because this is the uncertainty band used to detect the ground plane.

In the experiments we use a Pentium IV 1.8 GHz PC. Depending on the number of Points in the disparity image and using VTK and other prototyping software tools, the calculation requires less than three seconds. A C implementation is expected to be up to 100 as fast as shown in ground floor tracking in [Bur02]. The stereo system was mounted at $h = 1280mm$ above the ground. For experiments 4 and 5 the camera was placed at $h = 950mm$ above the ground plane to test the sensitivity to the viewing angle. It turned out that detection is reliable as long as the table surface can be partially seen. Detection of the chair in experiments 3 and 4 shows no difference in confidence of detection or height and size estimation. Only viewing the table rim from the height of the table is not sufficient for reliable detection. Consequently, only tables up to the height of the camera can be detected. The tilt angle θ of the camera system is about 68 degrees for all experiments and it deviates slightly due to floor unevenness.

Figure 39: Final result of table detection. The table area is clearly separated from data points at the same height in the background. For accuracy of the results see Table 3.

The experiments are compared to ground truth data, which was manually obtained. Table 3 summarises the ground truth and the estimated values for the experiments. The example presented in the previous Sections is given as Experiment 1. Four more experiments, #2-5, have been conducted with different arrangements. Fig. 40 shows views from the left camera. Fig. 41 shows the tables detected.

Table 3: Accuracy of table localization. Units are millimeters.

Exp. #	Object	Ground truth	Estimation	Deviation	%
1	Table	720	724	4	0.56
2	Table	455	476	21	4.62
3	Chair	475	495	20	4.21
4	Chair	475	484	9	1.89
5	Chair	475	480	5	1.05
6	Table	455	467	12	2.64

The table shows that height estimation if accurate within $20mm$ or better than 5%, which justifies the use of such loose thresholds as 20 degrees or $10cm$ used in the detection process. The distance to the front rim of the table was also assessed, however, the process to obtain ground truth is not reliable, since the camera internal coordinate system is not accessible. These measurements have shown that table distance was about $1.5m$ and was estimated to within $100mm$.

Figure 40: Left images of four further experimental set-ups referring to experiments #2-5 in Table 3.

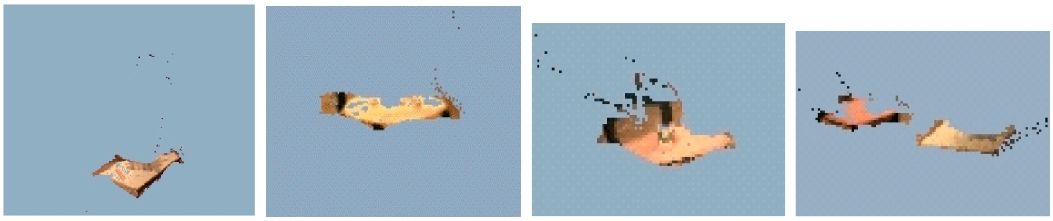


Figure 41: Disparity image for the images in Fig. 40 indicating the tables detected.

4.5 Structural entities from range data

As indicated in section 2.7, the intended features extracted from range data for furniture recognition are horizontal and vertical planar regions. Principally, these features are also appropriate to describe the floor plane, the ceiling as well as walls. The most important of these three to determine the room layout are walls. In order to determine wall points in a range image, Wulf et al propose in [OWW04] to choose the point of each scan column of the range image with the largest distance to the sensor. As walls build the boundary of a closed indoor scene, these points are most probably wall points. The resulting "2D scan" is only disturbed by open doors, windows and objects that cover the wall completely.

5 Conclusion

We have presented several methods to extract basic cues for structural decomposition, developed in the course of the first year of the robots@home project. Here the main focus was on the detection of planar surfaces either from monocular or from stereo/trifocal images - representing strong cues suitable for further processing such as recognition of certain types of furniture (tables, cupboards), or doors, windows, or simply walls.

Preliminary experiments have shown promising results, however in many cases robustness was an issue, stemming from the purely bottom-up processing nature of the methods described. We plan to direct further research towards robust approaches and the fusion of the proposed modalities and cues in order to obtain stable results in a computationally efficient way.

In the future, we will investigate statistical learning approaches to directly learn visually meaningful, repetitive parts from large example sets without the need to identify and model structural decompositions into parts - avoiding the typical pitfall of relying on to general a-priori assumptions (or worse, heuristics) which do not hold in general settings. Assumptions like "a chair has four legs", or "a table has a flat surface in a certain height" are good examples - they are either hard to verify (clutter on the table), or not true in the general sense (chairs with less then, or without legs do indeed exist). Also, even if the structural model holds, the object's parts are not bound to exhibit a simple shape, and hence do not guarantee a low intra-class variability. Furthermore, it is not clear how function-based, structural models (not just their parameterisation) would be fully learned from data, as the verification or reasoning about a parts structural function (a leg gives stable support to the table surface) would require the agent to interact with the object. Simple geometric or topologic relations will not suffice to tell if a certain part really provides stable support - a single, off-centre leg can support a table, as long as it has a flat supporting socket and is properly screwed on.

6 Appendix

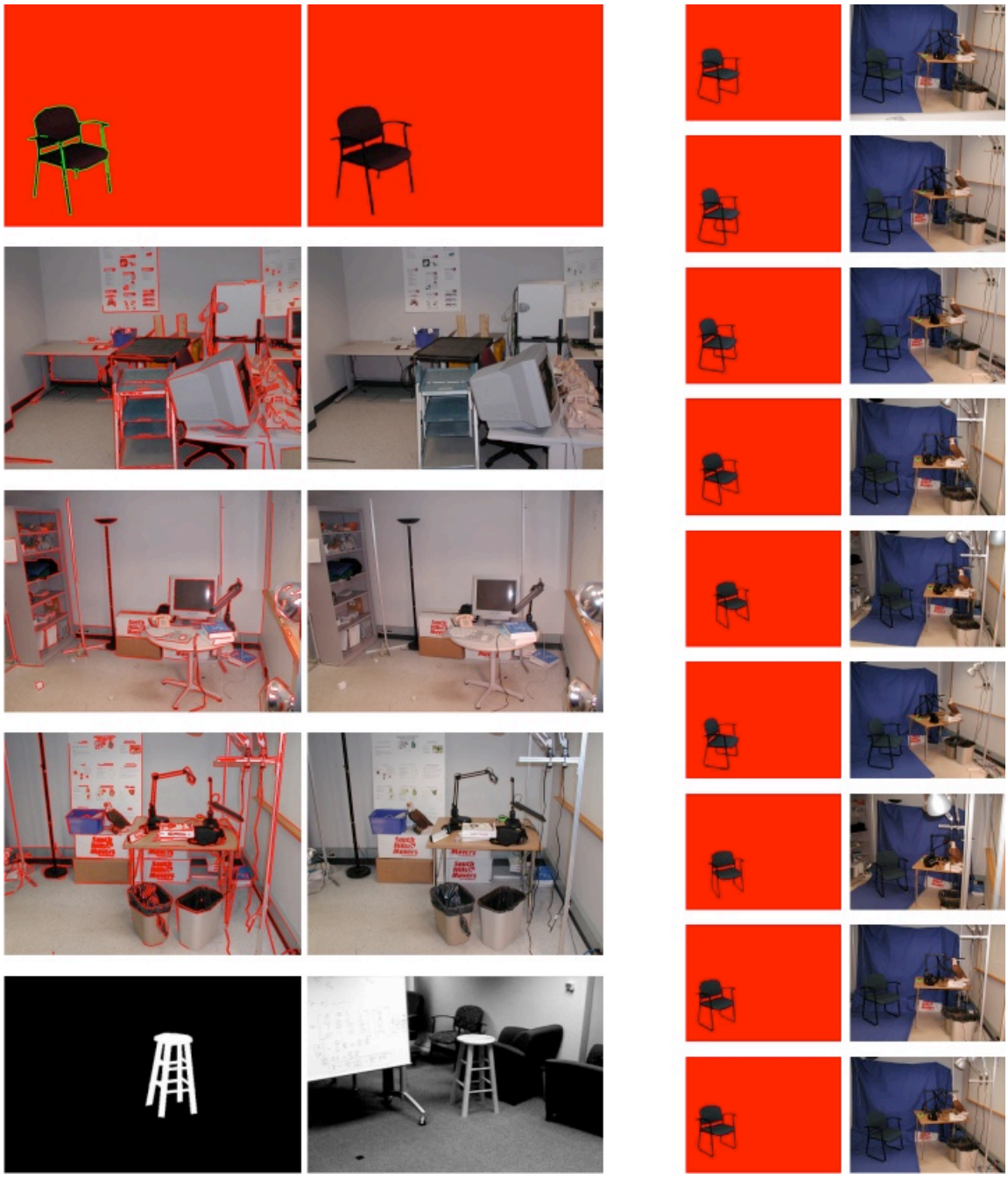


Figure 42: Example images from the WORD database.

Figure 43: Example images and ground-truth masks from the LabelMe database.

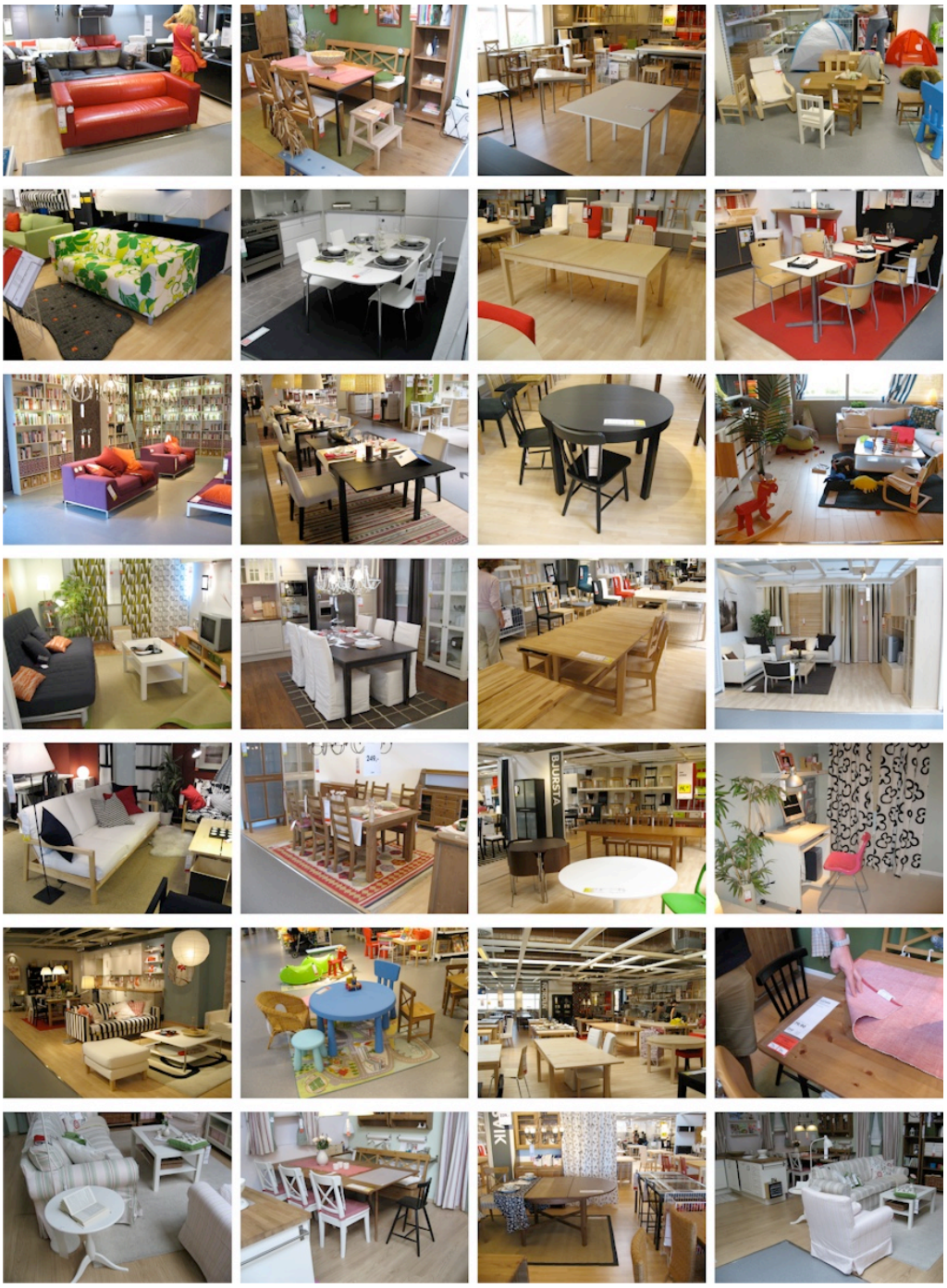


Figure 44: Exampleset of the single image collection at IKEA.

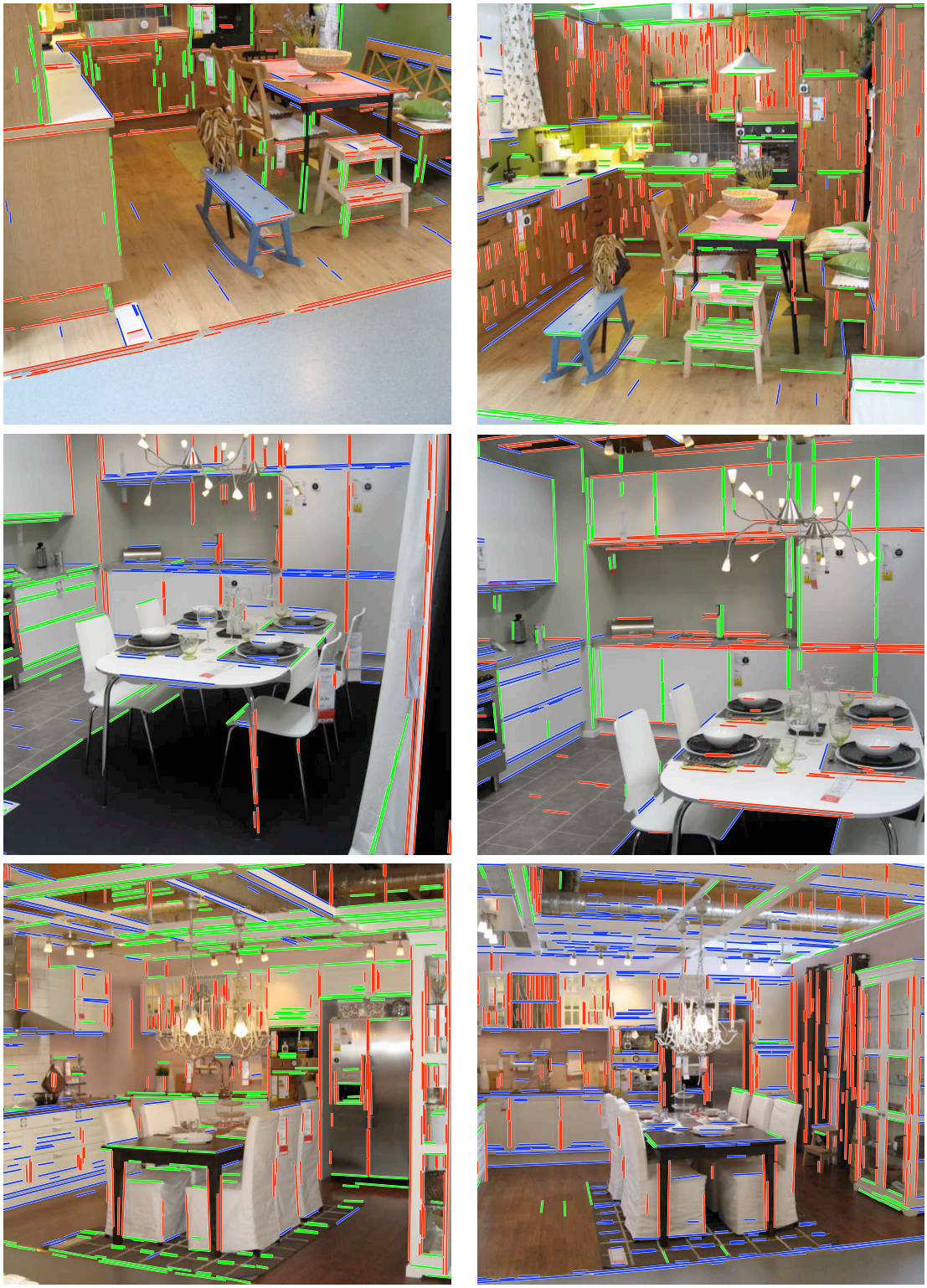


Figure 45: Vanishing point detection results for Ikea images. Lines corresponding to three orthogonal vanishing directions colour coded in red, green, and blue.

Figure 46: Vanishing point detection results for Ikea images.

Figure 47: Vanishing point detection results for Legrand images.

Figure 48: Rectangle detection results. First row: Original image and complete rectangles detected in the dominant orthogonal planes. Second to last row: Closed incomplete rectangles (left column) and closed U-shapes (right column).

Figure 49: Exampleset of the stereo sequences at our laboratory.

Figure 50: Exampleset of the autonomous stereo sequence at IKEA.

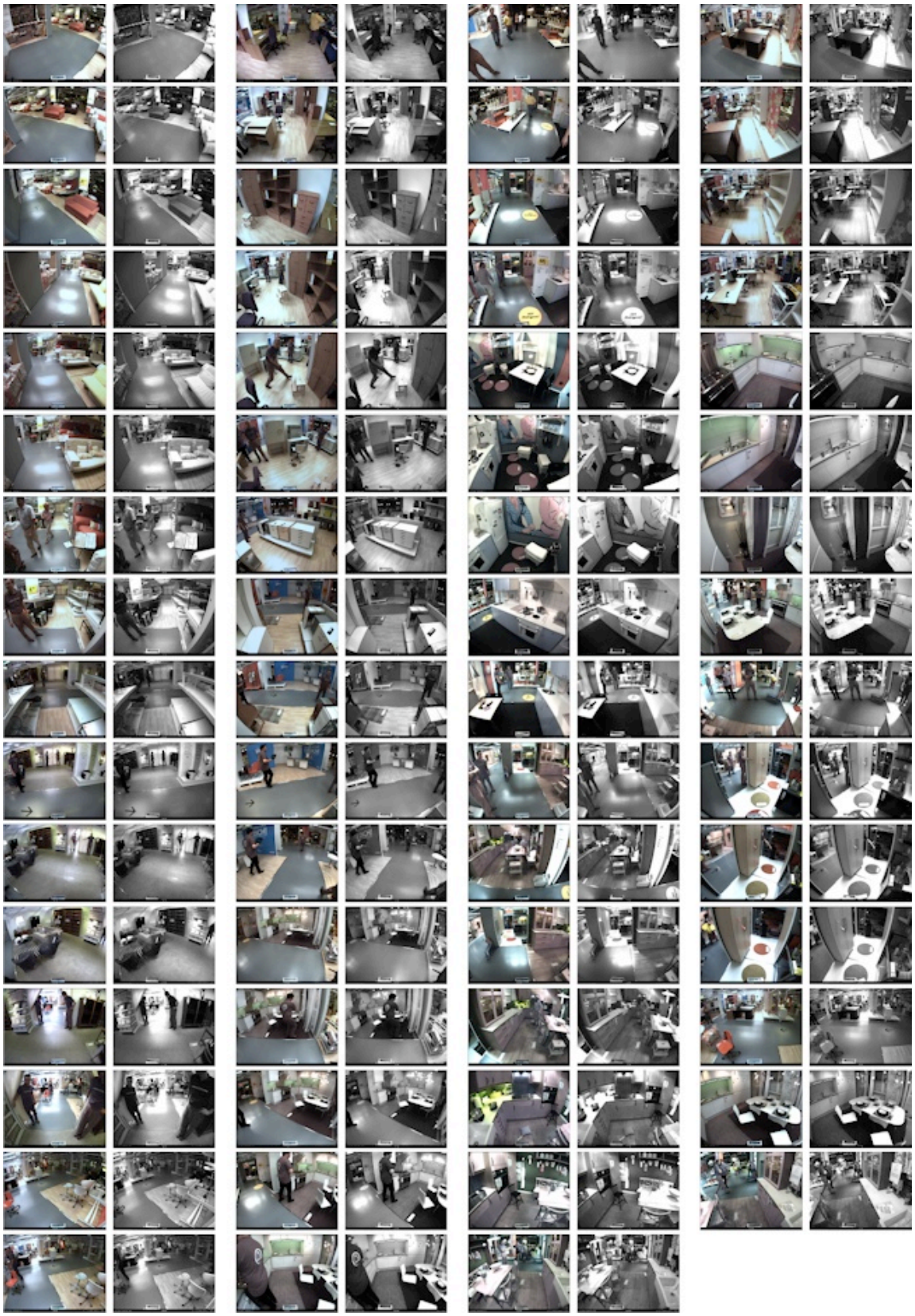


Figure 51: Exampleset of one of the stereo sequences at IKEA.

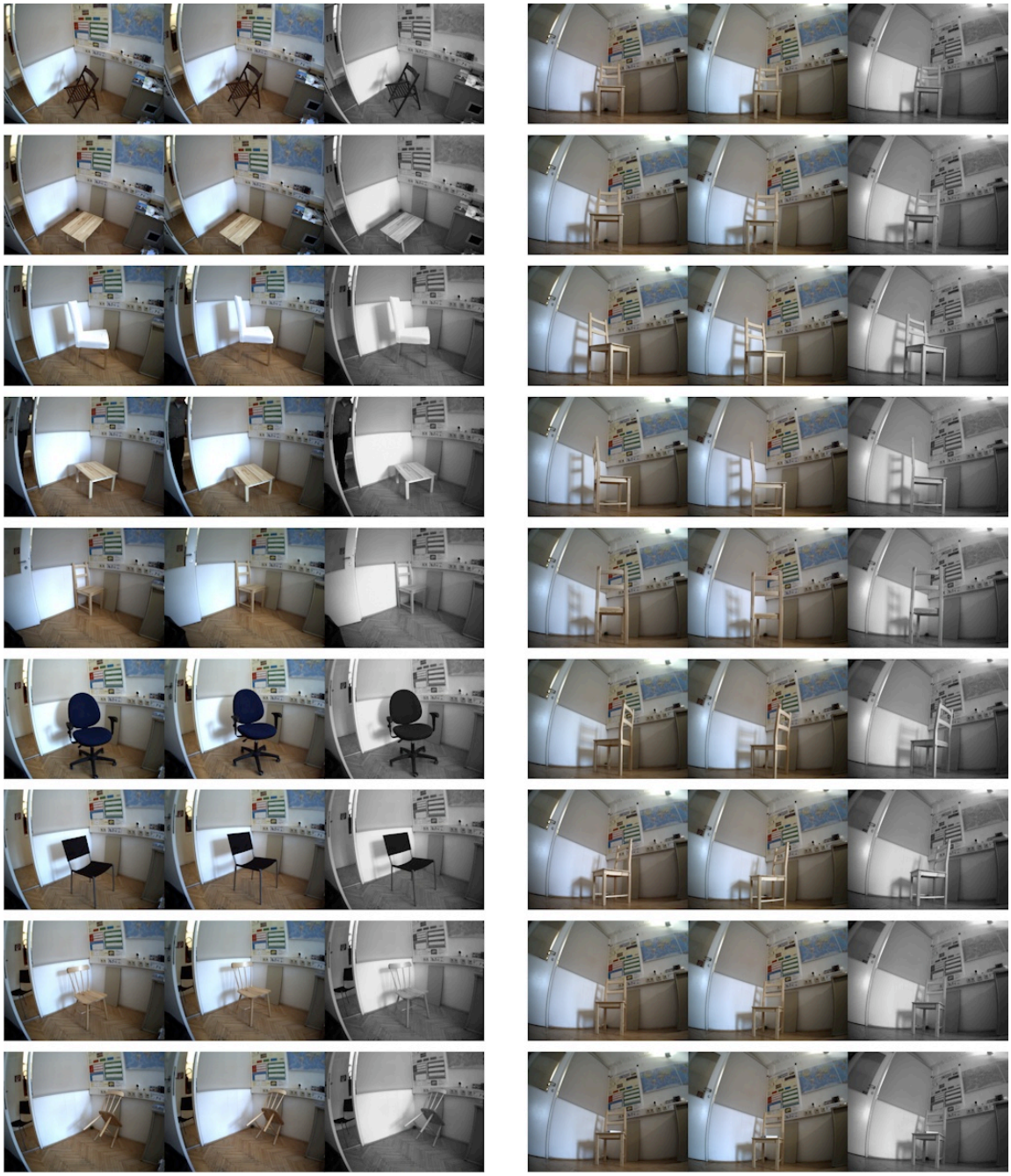


Figure 52: Examples of the trinocular stereo image collection.

Figure 53: Examples of the chair image collection.

References

[AGLF05] D.G. Aguilera, J. Gomez-Lahoz, and J. Finat. A new method for vanishing points detection in 3d reconstruction from a single view. *The Intl. Archives of photogrammetry, remote sensing and spatial information sciences: Virtual Reconstruction and Visualization of Complex Architectures*, 26, 2005.

[AT00] M. Antone and S. Teller. Automatic recovery of relative camera rotations for urban scenes. In *Proc., Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2282–2289, 2000.

[BBFVG05] H. Bay, H. Bay, V. Ferraris, and L. Van Gool. Wide-baseline stereo matching with line segments. In V. Ferraris, editor, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2005*, volume 1, pages 329–336 vol. 1, 2005.

[BO91] B. Brillault and O'Mahony. New method for vanishing point detection. *Computer Vision, Graphics, and Image Processing*, 54(2):289–300, 1991.

[Bou07] Jean-Yves Bouguet. Camera calibration toolbox for matlab, 2007.

[Bro03] D.; Hager G.D.: Brown, M.Z.; Burschka. *Advances in computational stereo;.* IEEE Transactions on Pattern Analysis and Machine Intelligence 25(8), pp. 993-1008, 2003.

[Bur02] Hager G.: Burschka, D. *Scene Classification from Dense Disparity Maps in Indoor Environments.* IEEE ICPR Int. Conf. on Pattern Recognition, Quebec, 2002.

[Ch04] Owen Carmichael and Martial hebert. Word: Wiry object recognition database. rope.ucdavis.ed/õwenc/word.htm, January 2004. Carnegie Mellon University.

[CY03] James M. Coughlan and Alan L. Yuille. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Computation*, 15(5):1063–1088, 2003.

[EVGW+07] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html, 2007.

[Fau93] Olivier Faugeras. *Three-Dimensional Computer Vision.* MIT Press, 1993.

[FH04] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *Int. Journal of Computer Vision (IJCV)*, 59(2):167–181, 2004.

[Gon02] Woods R.E.: Gonzalez, R.C. *Digital Image Processing; 2nd ed.* Prentice Hall, Upper Saddle River, NJ, 2002.

[HEH05] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Geometric context from a single image. In *Tenth IEEE International Conference on Computer Vision (ICCV)*, pages 654–661, 2005.

[HEH06] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *Proc., Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2137–2144, 2006.

[HEH07] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *International Journal of Computer Vision (IJCV)*, 75(1), Oct. 2007.

[HLD07] J.B. Hayet, F. Lerasle, and M. Devy. Visual landmarks detection and recognition for mobile robot navigation. *Image and Vision Computing*, 25(8):1341–1351, August 2007.

[HSEH07] D. Hoem, A.N. Stein, A.A. Efros, and M. Hebert. Recovering occlusion boundaries from a single image. In *International Conference on Computer Vision (ICCV)*, 2007.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in computer vision*. Cambridge University Press, 2003.

[HZ04] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[Kah07] Timo Kahlman. *Range Imaging Metrology: Investigation, Calibration and Development*. PhD thesis, ETH Zurich, 2007.

[Köt03] Ulrich Köthe. Edge and junction detection with an improved structure tensor. In *Proc., Symposium of German Association for Pattern Recognition (DAGM)*, pages 25–32, 2003.

[Kov] Peter Kovesi. Matlab and octave functions for computer vision and image processing. School of Computer Science & Software Engineering, University of Western Australia.

[KS05] Kenichi Kanatani and Yasuyuki Sugaya. Statistical optimization for 3-d reconstruction from a single view. *IEICE Transactions on Information and Systems*, E88-D(10):2260–2268, 2005.

[KZ02a] Jana Košecká and Wei Zhang. Efficient computation of vanishing points. In *Proc., Intl. Conference on Robotics and Automation (ICRA)*, pages 223–228, 2002.

[KZ02b] Jana Košecká and Wei Zhang. Video compass. In *Proc., European Conference on Computer Vision (ECCV)*, pages 476–490, 2002.

[KZ05] Jana Košecká and Wei Zhang. Extraction, matching and pose recovery based on dominant rectangular structures. *Computer Vision and Image Understanding (CVIU)*, 100(3):174–293, 2005.

[Lie01] David Liebowitz. *Camera Calibration and Reconstruction of Geometry from Images*. PhD thesis, University of Oxford, 2001.

[LMS+06] John Lim, Chris McCarthy, David Shaw, Luke Cole, and Nick Barnes. Insect inspired robots. In *Proc., Australasian Conference on Robotics and Automation (ACRA)*, 2006.

[Low87] David Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.

[MA84] Michael J. Magee and Jake K. Aggarwal. Determining vanishing points from perspective images. *Computer Vision, Graphics, and Image Processing*, 26(2):256–267, 1984.

[MP07] Branislav Micusik and Tomas Pajdla. Multi-label image segmentation via max-sum solver. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[MWK08] Branislav Micusik, Horst Wildenauer, and Jana Košecká. Detection and matching of rectilinear structures. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[OWW04] Henrik I. Christensen Oliver Wulf, Kai O. Arras and Bernardo Wagner. 2d mapping of cluttered indoor environments by means of 3d perception. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA*, 2004.

[PB05] R. Pflugfelder and H. Bischof. Online auto-calibration in man-made worlds. In *Proc., Digital Image Computing: Techniques and Applications*, pages 519–526, 2005.

[RES+06] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[RLK05] M. Rous, H. Lupschen, and K.F. Kraiss. Vision-based indoor scene analysis for natural landmark detection. In *Proc., Intl. Conference on Robotics and Automation (ICRA)*, pages 4642–4647, 2005.

[RM03] X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings of the 9th Int. Conf. on Computer Vision*, volume 1, pages 10–17, 2003.

[Rot02] Carsten Rother. A new approach to vanishing point detection in architectural environments. *Image Vision Computing (IVC)*, 20(9-10):647–655, 2002.

[RTMF08] Bryan Russell, Antonio Torralba, Kevin Murphy, and William Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision (IJCV), to appear*, 2008.

[RW95] P.L. Rosin and A.W. West. Nonparametric segmentation of curves into various representations. *Pattern Analysis and Machine Intelligence*, 17(12):1140–1153, 1995.

[SMP05] Tomáš Svoboda, Daniel Martinec, and Tomáš Pajdla. A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422, August 2005.

[SSZ97] C. Schmid, C. Schmid, and A. Zisserman. Automatic line matching across views. In A. Zisserman, editor, *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 666–671, 1997.

[Wer07] T. Werner. A linear programming approach to Max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 29(7):1165–1179, 2007.

[WMV07] Horst Wildenauer, Branislav Micusik, and Markus Vincze. Efficient texture representation using multi-scale regions. In *Asian Conference on Computer Vision (ACCV)*, 2007.

[WV07] Horst Wildenauer and Markus Vincze. Vanishing point detection in complex man-made worlds. In *Proc., Intl. Conference on Image Analysis and Processing (ICIAP)*, 2007.

[YFW05] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. on Information Theory*, 51(7):2282–2312, 2005.

[Zil07] Michael Zillich. Incremental indexing for parameter-free perceptual grouping. In *Proceedings of the 31st Workshop of the Austrian Association for Pattern Recognition*, pages 25–32, 2007.