Dynamic composition of tracking primitives for interactive vision-guided navigation

Darius Burschka and Gregory Hager

Johns Hopkins University, Baltimore, USA

ABSTRACT

We present a system architecture for robust target following with a mobile robot. The system is based on tracking multiple cues in binocular stereo images using the XVision toolkit [1]. Fusion of complementary information in the images, including texture, color and depth, combined with a fast optimized processing reduces the possibility of loosing the tracked object in a dynamic scene with several moving targets on intersecting paths.

The presented system is capable of detecting objects obstructing its way as well as gaps. It supports application in more cluttered terrain, where a wheel drive of mobile robot cannot take the same path as a walking person.

We describe the basic principles of the fast feature extraction and tracking in the luminance, chrominance and disparity domain. The optimized tracking algorithms compensate for illumination variations and perspective distortions as already presented in our previous publications about the XVision system.

Keywords: vision-based navigation, 3D tracking, color tracking

1. INTRODUCTION

Tracking is an essential task in mobile systems. It is used to perform a variety of tasks including localization, obstacle avoidance, surveillance and gesture recognition. A common problem in most applications of mobile systems is to keep the vehicle on a pre-defined path. This path may be a corridor through a factory to transport parts from one machine to another, or a route through a building to give a pre-specified tour [2], or it may be a known path between offices in case of a courier robot [3]. Several systems have been proposed to solve this problem, most of which operate based on maps [4], [5], [6], [7] or based on localization from artificial landmarks in the environment [3].

Map-based systems use a stored two or three-dimensional representation of the environment together with sensing to provide such a reference. However, it is not clear that building a metrically accurate map is in fact necessary for navigation tasks which only involve following the same path continuously. Another approach would be to use no prior information, but rather to generate the control signals directly from only currently sensed data [8]. In this case no path specification at all is possible. For this second field of applications tracking is essential.

The main problem in tracking applications on mobile systems are the changing light and geometric conditions. A static or quasi-static camera allows a-priori optimization of the tracking primitives for robust operation. This task proves to be more complex on mobile systems, where changing light conditions and environment complexity do not allow a fixed set of tracking clues.

Usually, the restricted resources on mobile systems limit the number of tracking tasks to be run in parallel. An optimal set of them needs to be chosen depending on the current situation. This choice is only possible, if a global error function can be found for all tracking primitives that makes it possible to compare their results.

We structure this paper as follows. In section 2 we introduce the camera model of our system. In section 3, we give a global system description (section 3.1) followed by a description of the processing in the single

Further author information:
D. Burschka: E-mail: burschka@cs.jhu.edu, Computational Interaction and Robotics Laboratory (CIRL)
G. Hager: E-mail: hager@cs.jhu.edu, Computational Interaction and Robotics Laboratory (CIRL)

system layers. We present a method for calculating the quality of the tracking result for the single primitives in section 3.4.3. The results section (section 4) presents the experimental tracking results in real system applications. We conclude in section 5 with a few remarks about the system and a description of future work.

2. CAMERA MODEL

We use in our system a binocular stereo camera system with two cameras mounted in a distance B from each other. The focal lengths of the lenses are f_L, f_R. The sensor image is organized as a matrix with the horizontal and vertical coordinates (u, v) originating in the middle of the image (Fig. 1). The origin of the world coordinate system (x, y, z) is in the optical center of the left camera that is used as a reference image in the entire processing. The image planes of the cameras are parallel to each other.

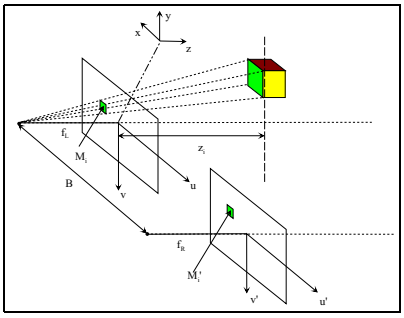


Figure 1: Coordinate systems and dimensions used in this paper.

In the following text we will call the tracking primitives *features*. Possible *features* in the system are lines, corner points, color regions, etc. Each feature is specified by its middle point M_i, horizontal and vertical extension d_u, d_v and distance from the image plane z_i, if available.

3. APPROACH

In this paper we present a tracking system that choses automatically a best set of tracking tokens to fulfill the task specified in the coordination layer (Fig. 2). We introduce a general tracking system that can dynamically compose an optimal set of tracking primitives for a given task and adapting to changing light conditions and environments. We will motivate it with an example of a robot following an object through different parts of our lab under different light conditions and densities of the surrounding objects.

3.1. System description

The entire tracking process is subdivided into four layers in our system:

- **Physical Sensor Layer** - this layer is responsible for image acquisition from the physical sensor into the system memory and contains the interfaces to the actual hardware drivers;
- **Image Processing Layer** - this layer is responsible for filtering and extraction of relevant information from the raw sensor image. We distinguish two categories of image processing as depicted in Fig. 2: feature extraction, where the image content is abstracted to derive information about region boundaries in form of corner points, lines and curves, and domain conversion, where the image content is just transformed to a desired representation, like hue, gray-scale, disparity;
- **Feature Identification Layer** - this layer identifies the position of the tracked features in the current image frame and passes their position together with a quality value to the tracking module;

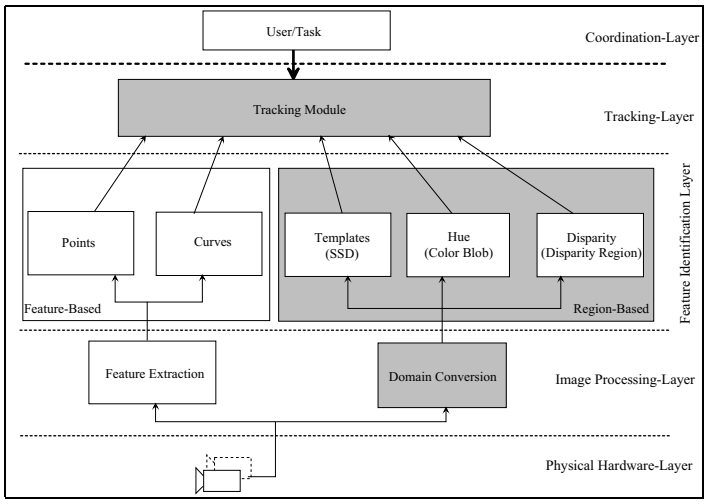


Figure 2: Tracking System Hierarchy.

- **Tracking Layer** - this layer maintains the state of the tracked object. It is responsible for filtering, predictions that are required to follow the movement of the tracked object in the image. This layer decides, based on the quality value γ_i described in section 3.4.3, which features are appropriate in a given moment in time.
- **Coordination Layer** - this layer is not part of the tracking system, but it represents the interface to the real application.

In this paper we will discuss the implementation of the gray-shaded modules of the system. We will show with an example of color blob and disparity tracking, how tracking modules can be dynamically composed.

3.2. Physical Sensor Layer

As already mentioned in the global description (section 3.1) the purpose of this layer is robust and efficient image acquisition from the physical sensor into the system memory. In our implementation, we use ring structure for the frame buffer where consecutive frames are stored. Each frame is stored with a time-stamp referencing the point in time when the image was acquired.

3.3. Image Processing Layer

3.3.1. Color segmentation

The color images acquired from the camera can have different representations. While most graphics system favor the RGB color representation, the YUV color coding seems to be a better alternative for color processing. Fig. 3 depicts the relation between these two representations.

Our color segmentation uses the fact that the hue information Θ for a given surface P stays constant for all brightness values Y (Fig. 3). Therefore, in the YUV representation just the UV part is used to compute the hue. The RGB information needs to be split with into its YUV components. The segmentation subdivides the color circle in the UV plane into sectors

$$\mathcal{C}_j =] \, (i+1) \cdot \Delta\Theta; i \cdot \Delta\Theta \,] \tag{1}$$

3.3.2. Disparity segmentation

In our system we use dense disparity images from a correlation based stereo reconstruction algorithm as the source of data for processing [9].

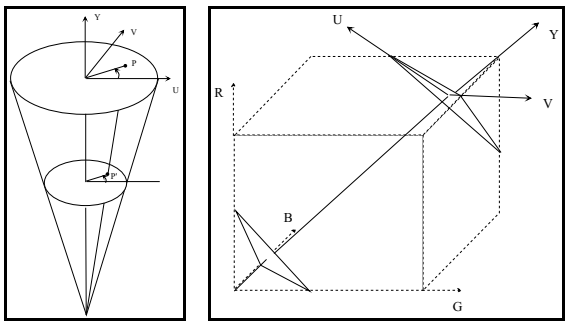


Figure 3: Different image representations: (left) YUV, (right) RGB color coding.

3.4. Feature Identification Layer

In this paper we want to concentrate on two feature identification processes: color blobs and disparity regions. An example of each of the regions in the corresponding image domain is shown in Fig. 4.

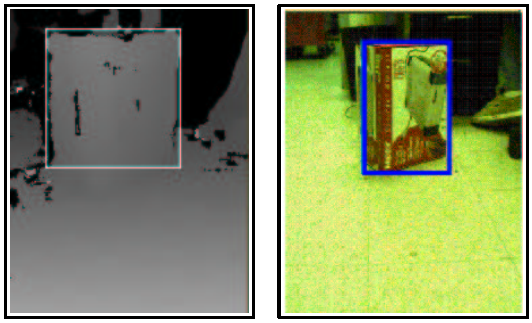


Figure 4: Region selection in disparity and color image space.

The Feature Identification Layer needs to provide three basic functionalities that are subsequently required by the Tracking Layer: initial feature selection, evaluation of uniqueness, and localization in the image.

3.4.1. Initial feature selection

A region $\mathcal{R}_i$ in the image represents a set of pixels with a similar property in the chosen image domain. In case of a color image it can be the hue range of the corresponding pixels or in case of a disparity image it can be a disparity range describing a cluster of points in space.

The goal is to identify a unique object that can be detected robustly in consecutive images. The uniqueness of the object is in our case defined based on the following criteria.

Compactness in the object space. The tracked region in our system represents a compact cluster spanning a range in the given domain. We require that the cluster to be continuous in the given domain to ensure that the region property is preserved during the tracking process. In real images areas on an object may not be detected correctly due to texture on the surface in this area. Therefore, we analyse the histogram of the image to specify the range of the tracked object in the domain. This search can be done for the entire image, which corresponds to an automatic landmark selection, or it can be restricted to a local area of the image specified by the user.

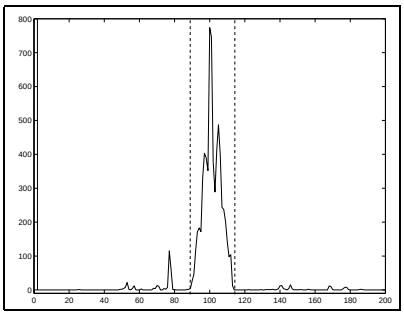


Figure 5: Example of a landmark selection in a disparity image.

Fig. 5 depicts the initial target selection in the disparity image shown in Fig. 4. In this case the nearest significant object was supposed to be selected. From the histogram the range of the peak $[h_l; h_u]$ was estimated as shown with the dashed lines.

In this example we require the distances z_i for all points to be in the estimated disparity range between the dashed lines.

$$h_l < \frac{B \cdot f_L}{z_i \cdot s_x} < h_u \tag{2}$$

The values B, f_L, z_i are already introduced in Fig. 1 and s_x represents the horizontal pixel-size of the camera chip. This is a basic stereo disparity equation discussed in [10] in more detail. This restriction may be sufficient to detect clusters in a local window, where only the tracked object needs to be segmented from the background, but it cannot be applied on the entire image. Objects in similar depth can appear in the image that should not be classified as members of the same cluster $\mathcal{R}_i$. Properties in the image space need to be taken into account.

Compactness in the image space. The extraction of a region in a given image domain requires uniformity of the imaged surfaces in this domain. In ideal case only pixels creating a continues area in the image should be selected. This assumption is often violated in real images due to noise in the sensor, texture and shape of the surface, and light conditions. To compensate for these errors we allow gaps between the neighboring pixels.

We require that the distance between neighboring image elements (pixels) p_x in the considered cluster should not exceed a given threshold ϵ_c.

$$|p_i - p_j| < \epsilon_c \tag{3}$$

3.4.2. Identification in the disparity domain

The initial identification in the disparity domain needs an additional processing step. The problem is to extract single standing clusters representing objects from continuous disparity images. An example is shown in Fig. 6. This image shows several objects, which are classified correctly, with different heights.

Typically, the floor connects the entire image to one large cluster, making any kind of segmentation difficult if not impossible. Since the floor is not interesting for further processing, our first step is to remove it from the input data.

Geometrical constraints. In stereo data processing, the disparity value d_p in the image depends solely on the distance z_p between the imaged point P and the image plane of the sensor (Fig. 7). In case of a camera tilted at an angle Θ against the ground plane and pointing at an empty floor, each pixel in a horizontal image row has the same disparity

$$d_p = \frac{B}{z_p} \cdot \frac{f}{p_x}, \tag{4}$$

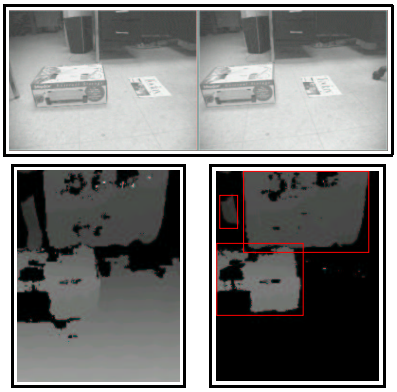


Figure 6: Detected three obstacles (right) from a pair of images (top) in the dense disparity image (left).

because z_p is independent of the horizontal coordinate in the image. B is the distance between the two cameras of the stereo system, f is the focal length of the camera, which is used as base for the reconstruction (in our case the left camera), and p_x is the horizontal pixel-size of the camera chip.

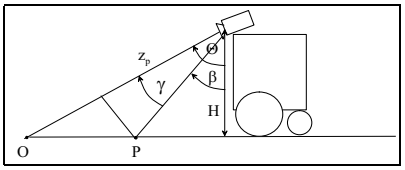


Figure 7: Geometrical consideration for expected disparity in an image.

The estimates for d_p or z_p in equation (4) come directly from the image. This is done in an on-line re-calibration process, which is described in the next section.

Estimation of the ground plane. In indoor environments systems usually operate on flat surfaces. These surfaces can be estimated from the sparse information available in the images.

The internal camera parameters and the orientation of the cameras to each other are estimated in an off-line calibration process [11], [12]. The re-calibration procedure running on-line in each reconstruction step estimates the orientation of the camera system with respect to the ground plane μ (Fig. 8). Basically, the presented calibration process estimates the rotation between the coordinate systems of the camera (u, v, e) and the world (x, y, z).

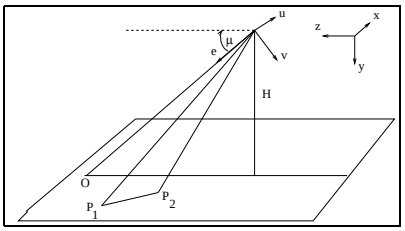


Figure 8: Calibration of Θ and H from two arbitrary points P_1, P_2.

The calibration values can be calculated based on the reconstructed position of two points P_i, P_j, which are part of the ground plane $\mathcal{P}$. The stereo reconstruction process reconstructs the 3D position of each point

in the image. Since the system is supposed to be used for collision avoidance, the probability is high that the bottom rows of the image contain, at least partially, the ground plane. A histogram over the entire image row is calculated in 10 different rows in the bottom part of the image and the peak values are used to estimate disparity value d_p for this row. A pixel in each row with exactly this disparity value is used to estimate the coordinates of the point P_x in the coordinate system (u, v, e) of the camera system.

The angle μ_k can be calculated using the scalar product of the normalized vector $\overline{P_i P_j}$ between any two of the ten extracted points and the normalized vector along the z-axis $\overline{z_0}$.

$$\begin{aligned} n &= |P_2 - P_1| \\ &= \sqrt{(u_2-u_1)^2 + (v_2-v_1)^2 + (e_2-e_1)^2} \\ \mu_k &= \arccos\left[\frac{1}{n}\begin{pmatrix} u_2-u_1 \\ v_2-v_1 \\ e_2-e_1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right] \\ &= \arccos\frac{|e_2-e_1|}{|P_2-P_1|} \end{aligned} \tag{5}$$

The set $\{\mu_k\}$ is used to vote for the estimated angle μ_{est}. The RANSAC [13] method can be used to estimate a valid set $\mathcal{S}$ of points reconstructing μ_{est}. The calibration value Θ can be calculated using μ_{est} as

$$\Theta = \frac{\pi}{2} - \mu_{est} = \frac{\pi}{2} - \arccos\frac{|e_j - e_i|}{|P_j - P_i|}. \tag{6}$$

The height of the camera system H can be estimated from the scalar product of the vector $\overline{P_x}$ with the z-axis expressed in the coordinate system of the camera

$$\begin{aligned} H_x &= \begin{pmatrix} u_x \\ v_x \\ e_x \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \cos\mu_{est} \\ \sin\mu_{est} \end{pmatrix} \\ &= v_x \cdot \cos\mu_{est} + e_x \cdot \sin\mu_{est} \\ \Rightarrow \quad H &= \frac{1}{|\mathcal{S}|} \cdot \sum_{x\in\mathcal{S}} H_x. \end{aligned} \tag{7}$$

We have included a "sanity" check in our system that verifies the computed values to catch outliers. If the calculated height changes differs significantly $\Delta H > 10cm$ from the initially estimated value then the current calibration is rejected.

Prediction of the disparity for the ground plane. The parameter z_p can be calculated from the geometrical values depicted in Fig. 7 as

$$\begin{aligned} z_p &= \frac{H \cdot \cos\gamma}{\cos\beta}, \\ with \qquad \beta &= \Theta + \gamma \ \wedge \ \gamma = \arctan\frac{v_p \cdot p_y}{f}, \end{aligned} \tag{8}$$

where v_p is the vertical pixel coordinate in the image relative to the optical center, pointing down (Fig. 8 and p_y is the vertical pixel-size of the camera. The angle γ is the vertical angle in the camera image between the optical axis and the current line v_p.

Using the equations (4),(8) we can formulate the equation for the expected disparity value for a given line v_p to be

$$\begin{aligned} d_p &= \frac{B \cdot f}{H \cdot p_x} \cdot (\cos\Theta - \sin\Theta \cdot \tan\gamma) \\ &= \frac{B \cdot f}{H \cdot p_x} \cdot \left(\cos\Theta - \sin\Theta \cdot \frac{v_p \cdot p_y}{f}\right). \end{aligned} \tag{9}$$

Using the equation (9) the initial disparity image is processed to remove the supporting plane.

3.4.3. Evaluation of the uniqueness factor γ_i

The quality of the tracked feature can be defined based on two factors:

- **distance ϵ to similar regions** in the scene. The similarity of the regions is defined based on equations (2), (3).

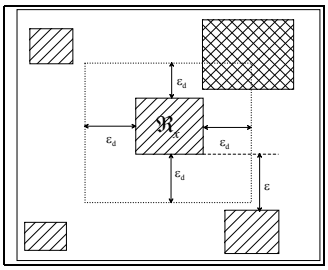


Figure 9: Region uniqueness.

All image elements that are in a distance smaller than ϵ_c are merged to a single region (equation (3)). In the current application the value ϵ_d (Fig. 9) is chosen to

$$\epsilon_d = 10 \cdot \epsilon_c. \tag{10}$$

The distance dependent uniqueness value γ_{iD} is estimated to

$$\gamma_{iD} = \min(1, \frac{\epsilon}{\epsilon_d}). \tag{11}$$

- **cue relevance** γ_{iC} in the region is defined as the percentage of the pixels within the tracked region $\mathcal{R}_x$ that were classified as belonging to the tracked object.

The resulting uniqueness value γ_i is defined by the lower of the both values γ_{iC} and γ_{iD}

$$\gamma_i = \min(\gamma_{iC}, \gamma_{iD}). \tag{12}$$

This definition reflects the fact that a flaw in one field cannot be compensated by good performance in the other one. The resulting value γ_i has a range between $[0; 1]$ with 1 being the best value.

The same can be defined for the SSD algorithm (Fig. 2). This definition of the uniqueness allows a robust choice between the region-based tracking algorithms.

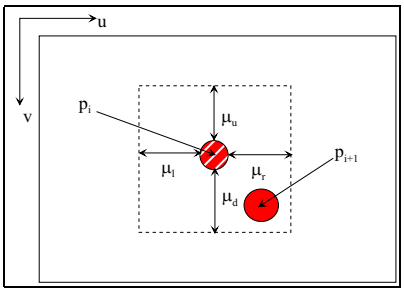


Figure 10: Definition of the subimage size based on p_i.

3.4.4. Localization of the feature in the subimage

Tracking in our implementation is based on re-localization of the tracked object in the image. The position of the object p_i at time t_i is used to define a search window in the subsequent frame acquired at time t_{i+1} (Fig. 10).

The $\mu_{x\in\{l,r,u,d\}}$ values represent the distances from the tracked region in all four directions (Fig. 10). In the initial step and in case the speed of the tracked region is not estimated (section 3.5), all $\mu_{x\in\{l,r,u,d\}}$ values in Fig. 10 are set to the same value, which needs to be large enough to keep the tracked object inside of the defined subimage. We set in this case $\mu_x = \epsilon_d$ (see equation (10)). If the speed of the object with the size r_u, r_v in the camera image is known (s_u, s_v) then the size of the sub-window (Fig. 10) can be estimated to be

$$\begin{aligned} \mu_l &= p_{iu} - \epsilon_d - s_u, \quad \mu_r = p_{iu} + \epsilon_d + s_u \\ \mu_d &= p_{iv} + \epsilon_d + s_v, \quad \mu_u = p_{iv} - \epsilon_d - s_u \end{aligned} \tag{13}$$

The processing in the *Image Processing Layer* (section 3.3) can be limited to the calculated region. The segmented image is clustered into regions and the region with the highest γ_i (section 3.4.3) is chosen as the result of the current localization step.

3.5. Tracking Layer

The tracking module maintains the state of the tracked object. This state needs to contain the following domain-specific properties $\mathcal{E}_s$:

- **position in the image** p_i - the position of the middle point of the tracked feature;
- **size of the region** r_u, r_v - the (u,v)-extension of the object in the image;
- **range in the current domain** d_{min}, d_{max} - the hue or disparity range of the tracked object;
- **shape in the image** ν - it describes the ratio of width to height in the image;
- **compactness of the region** q - it describes the percentage of the pixels that are in the valid range d_{min}, d_{max};
- **uniqueness** γ_i - the uniqueness of the region in the given image domain.

The robustness of the tracking process can be increased by estimating additional state values that are independent of the image domain. They describe global properties $\mathcal{E}_g$ of the actual object:

- **speed in the image** s_u, s_v - the speed of the tracked region $\mathcal{R}_x$ in the image;

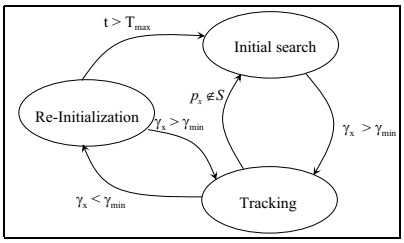


Figure 11: State transitions in the tracking process.

- **state in the object space** - the translation and rotation of the real object estimated based on, e.g., Kalman estimation [14] techniques.

The tracking process consists of three states, initial search, tracking, and re-initialization (Fig. 11), which are chosen based on the value of the current γ_i.

In the ***initialization state*** a unique region is selected. The initial state values are stored for all employed tracking cues. This step is necessary to re-initialize the tracker later based on the information from other cues. Fig. 4 depicts an example where, based on the segmentation results in the disparity domain, state information for the box in the color domain was extracted. In this example the disparity domain had a γ_i value of 0.95 compared to 0.24 in the color domain.

Once the appropriate region and method are selected, the system changes to the ***tracking state***. In this state, the active module in the *Feature Identification Layer* is triggered to estimate the current position p_i and the uniqueness value γ_i for the current step. The system stays in this state until one of two possible exceptions occur:

- boundary exception - the tracked region leaves the image $p_i \notin \mathcal{S}$. In this case the system switches back to the *initialization state*.
- uniqueness exception - the γ_i value drops below a threshold γ_{min}. In this case the system goes to the *re-initialization state*.

In the ***re-initialization state*** the other possible cues for the given subimage are queried to determine if they can provide a valid γ_i value. If this is the case and the shape ν and compactness q values match the initial estimates then the system switches back to the *tracking state* using the new cue. In the other case, the system predicts an estimated value for the region until T_{max} is reached and goes back to the *initialization state*.

4. RESULTS

In our experiments we used a Nomad Scout as a mobile robot with a PentiumIII@850MHz notebook running Linux-OS. The system was equipped with SRI's MEGA-D Megapixel Stereo Head with 8mm lenses. The cameras were mounted in a distance of 8.8cm from each other.

The typical tilt angle of the camera system during the experiments was $\Theta = 53°$. The system was mounted $H = 48.26cm$ above the ground. This configuration robustly detected obstacles in front of the robot while still allowing viewing up to 4m in front of the robot. In this configuration the system was running with a frequency of 11.2 Hz for the tracking.

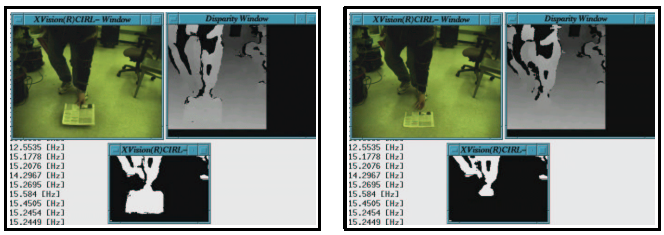


Figure 12: The newspaper is visible in the top, but it disappeared in the bottom image.

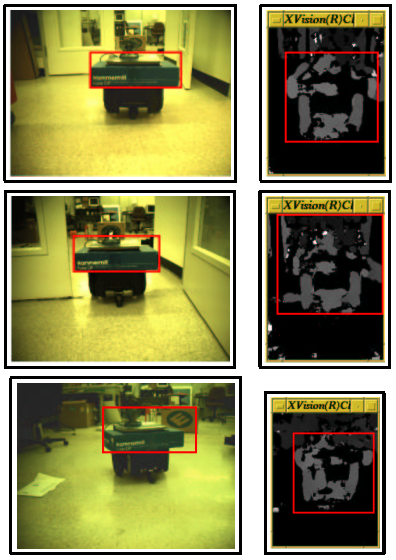
4.1. Quality of Ground Plane Detection

Ground plane suppression is fundamental for operation in the disparity domain. An example of suppression is shown in Fig. 12. It shows the resolution of the system, which is capable of distinguishing between the ground plane and objects as low as 1cm above the ground at a distance up to 2m. The newspaper disappears as an obstacle as soon as it lays flat on the ground.

Ground suppression was tested on different types of floor. We modified the tilt angle of the camera in a range $45° < \Theta < 70°$ in $5°$ steps. The number of pixels that could not be removed correctly was $0.6 \pm 0.01\%$ of the total number of pixels in the image. All these remaining pixels were incorrect depth estimations from the stereo algorithm.

4.2. Quality of Feature Identification

The algorithm was applied in a variety of situations and generated reliable results. A first example was already presented in Fig. 4 where, based on position of the region in one domain the corresponding region in the other one was selected. A few more examples are shown in the Fig. 13.



scene	disparity γ_i	color γ_i
before door	0.33	0.32
in door	0.22	0.33
behind door	0.42	0.30

Figure 13: Example of feature identifications during passing a door.

Fig. 13 shows the result of feature identifications for different scene types. In the first case due to poor texture and dark light conditions both tracker types return similar results. In this situation both tracker types could be used, but the disparity mode was chosen due to a slightly better γ_i value. In the door the neighborhood criterion seems to cause γ_i to drop for the disparity tracker. The color tracker, which shows constant values over the entire time is chosen. Behind the door the different light conditions and reduced complexity in the

distance range of the object raise the γ_i value for the disparity tracker significantly above the value of the color tracker.

5. CONCLUSIONS AND FUTURE WORK

We have presented a system that allows a dynamic composition of tracking primitives depending on the current quality value from the underlying identification process. It allows dynamic changes in the tracker composition depending on the current light conditions and environment complexity.

The system was tested on our mobile system *Goomba*, where it successfully tracked an object through the lab under changing environment complexity. The system was able to switch autonomously between color and disparity tracking while passing through narrow passages and it switched back to the more robust tracker once the scene complexity allowed it.

In the future we want to extend the set of modules used for tracking to allow a larger variety of composition possibilities that will allow more robust tracking and re-initialization.

ACKNOWLEDGMENTS

This work was supported by the DARPA MARS program, and by the NSF RHA program.

REFERENCES

1. G. Hager and K. Toyama, "The XVision System: A General-Purpose Substrate for Portable Real-Time Vision Applications," *Computer Vision and Image Understanding* **69**(1), pp. 23–37, 1995.
2. W. Burgard and A. Cremers and D. Fox and D. Hähnel and G. Lakemeyer and D. Schulz, "Experiences with an interactive museum guide-robot," *Artificial Intelligence* **I-53**, 2000.
3. J.M. Evans, "HelpMate: An Autonomous Mobile Robot Courier for Hospitals," *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, München* , pp. 1695–1700, 1994.
4. W. Burgard and D. Fox and D. Henning and T. Schmidt, "Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids," *Proc. of the 13th Nat. Conf. on Artificial Intelligence AAAI'96* , 1996.
5. D.J. Kriegmann and E. Triendl and T.O. Binford, "Stereo Vision and Navigation within Buildings for Mobile Robots," *IEEE Transactions on Robotics and Automation* , pp. 792–803, 1989.
6. X. Lebegue and J.K. Aggarwal, "Generation of Architectural CAD Models Using a Mobile Robot," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA'94)*, pp. 711–717, 1994.
7. S. Thurn, "An Approach to Learning Mobile Robot Navigation," in *Robotics And Autonomous Systems - special issue on Robot Learning*, 1995.
8. D. Burschka and G. Hager, "Vision-based control of mobile robots," *In Proc. of IEEE International Conference on Robotics and Automation* , pp. 1707–1713, May 2001.
9. T. A. Kanade and A. Yoshida and K. Oda and H. Kano and M. Tanaka, "A Stereo Machine for Video-rate Dense Depth Mapping and Its New Applications," in *Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society Press, 1996.
10. Oliver Faugeras, *Three-Dimensional Computer Vision*, Massachusetts Institute of Technology, The MIT Press, Cambridge, Massachusetts London, England, 1993.
11. Roger Y. Tsai, "A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses," *IEEE Transactions of Robotics and Automation* **RA-3**, pp. 323–344, Aug. 1987.
12. L. Iocchi and K. Konolige, "A Multiresolution Stereo Vision System for Mobile Robots," *AIIA (Italian AI Association) Workshop, Padova, Italy* , 1998.
13. M.A. Fischler and B.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography.," *Comm. ACM* **24**(6), pp. 381–395, 1981.
14. R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering, Transaction of the ASME* **83**, pp. 33–44, Mar. 1960.