A Robust Vision-Based Hover Control for ROV

Conference Paper · May 2008

DOI: 10.1109/OCEANSKOBE.2008.4531019 · Source: IEEE Xplore

Jochen Kerdels
FernUniversität in Hagen
36 PUBLICATIONS 131 CITATIONS

Jan Christian Albiez
Kraken Robotik GmbH, Bremen, Germany
107 PUBLICATIONS 1,096 CITATIONS

Frank Kirchner
Universität Bremen
313 PUBLICATIONS 2,696 CITATIONS

A Robust Vision-Based Hover Control for ROV

Jochen Kerdels
Underwater Robotics Department
DFKI-Lab Bremen
Germany, 28359 Bremen
Email: jochen.kerdels@dfki.de

Jan Albiez

Frank Kirchner

***Abstract*—The main field of application for small- and middle-class ROVs is the inspection of underwater structures or other objects of interest. Approaching such an object, one would want to hold a steady position in front of the object to study it in detail without having to concentrate on the control of the vehicle. This kind of "hover control" could be implemented by using an inertial measurement unit (IMU), but most of the small- and middle-class ROV do not have one. Furthermore, even the best IMUs tend to drift. On this account our approach, which is presented in this paper, uses video data to estimate the movements of the vehicle and uses this data to keep the vehicle hovering in front of a particular structure. The used vision algorithms are aimed at real world applications and are robust enough to handle various light and visibility conditions.**

I. Introduction

In recent years the number of available small- and middle-class remote operated vehicles (ROV) has risen significantly. The main field of their application is the inspection of underwater structures and other objects of interest. Due to their relatively small size and low weight, the vehicles are generally more prone to disturbances. Thus, holding a steady position in front of an object of interest can be a challenging task for the ROV pilot.

To assist the pilot in such a situation, an automatic station-keeping or "hover control" can be implemented. The system holds a steady position in front of an object while the pilot can study the object in detail without having to concentrate on the control of the vehicle. Such a system could be based on the data of an inertial measurement unit (IMU). But most of the small- and middle-class ROV today do not have an on-board IMU. Furthermore, even the best IMUs tend to drift. On this account we present a vision-based approach to automatic station-keeping of an ROV. The system uses video data to estimate the movements of the vehicle and thus keeps it hovering in front of a particular structure. Besides that every ROV is already equipped with a camera, drift cannot occur, as the measurements are absolute and not the integration of relative measurements.

The following section describes the vision algorithms which where used to estimate the movements of the vehicle. Afterwards the application of this approach on a small-class ROV, the LBV150 by seabotix, is shown. The paper closes with an outlook on future research on this topic in our research group.

II. Robust Vision

To accomplish the aforementioned motion estimation under various light and visibility conditions, it is essential to use an algorithm which is able to automatically adapt to these different conditions. A manual adjustment done by the operator would not be feasible, as the presented system is aimed at real world applications and should be easy to use by just switching it on or off without any additional parameters to be set. To satisfy this requirement, we introduce an extension to the well-known "Harris Feature Detector" which automatically specifies the needed parameters individually for every pixel of the particular image. The approach uses an efficient comparison of local and global entropy to decide which parameters should be used. This entropy-driven Harris detector chooses the most stable features in an image which are then subsequently tracked by the optical flow algorithm of Lucas and Kanade.

A. Harris Detector revisited

The well-known Harris detector [2] is an algorithm to detect salient features, such as corners, in an image. These salient features are good candidates to be tracked across several images of an image sequence. In our case, they can be used to determine the movement of the vehicle. The Harris detector is based on the idea of the earlier Moravec corner detector [1], which can be described as follows:

For each Pixel a local window $W_{x,y}$ is slightly moved into different directions while the changes in the intensity distribution are observed. If the local area is uniform, no or only little changes in the intensity distribution will occur. If there is an edge in the local area, only the movement orthogonal to the edge will have a large change in the intensity distribution. If there is a corner or a single spot, any movement will create a large change. Therefore, if the smallest change in the intensity distribution of all directions has a large value, then this indicates a single spot or a corner in the image.

This approach of Moravec has the drawback that a discrete number of directions are used. This reduces the response of edges and corners which are not aligned to any of the discrete directions. The Harris detector resolves this limitation by describing the intensity change E under a small translation (ϕ, θ) by

$$E(\phi, \theta) = (\phi, \theta)\, M\, (\phi, \theta)^T .$$

The matrix M describes the gradients of the local window $W_{x,y}$ in the image I around the point (x, y) by

$$M := \begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

with

$$A := \sum_{u,v \in W_{x,y}} \left(I_{u-1,v} - I_{u+1,v}\right)^2 * g_{u,v,x,y}$$

$$B := \sum_{u,v \in W_{x,y}} \left(I_{u,v-1} - I_{u,v+1}\right)^2 * g_{u,v,x,y}$$

$$C := \sum_{u,v \in W_{x,y}} \left(I_{u-1,v} - I_{u+1,v}\right) * \left(I_{u,v-1} - I_{u,v+1}\right) * g_{u,v,x,y}$$

and the weight function g

$$g_{u,v,x,y} := \exp \frac{-\left((u-x)^2 + (v-y)^2\right)}{2\sigma^2}.$$

The matrix M can be interpreted as the covariance matrix of the summed gradients of the local area in x and y direction. Let λ_1 and λ_2 be the eigenvalues of matrix M. For these eigenvalues considerations similar to those of the Moravec detector can be made. When both eigenvalues are small, no edges or corners are present in the local image area. If only one eigenvalue is big, there is an edge in the local area, as most of the gradients are oriented in one direction. If both eigenvalues are big, the gradients are oriented in different directions which hints at the existence of a corner in the local image area. In contrast to the Moravec detector, the Harris detector has a isotropic response and the use of a Gaussian weight function prevents sudden changes. To avoid the explicit calculation of the eigenvalues, as for example the comparable approach of Kanade and Tomasi [3] does, the trace and determinant of M can be used:

$$T(M) = \lambda_1 + \lambda_2 = A + B,$$

$$D(M) = \lambda_1 \lambda_2 = AB - C^2.$$

Using trace and determinant, a function R can be defined, which is positiv for corners, negativ for edges, and small for areas:

$$R := D - kT^2.$$

The parameter k defines the sensitivity of the detector. A bigger k reduces the value of R and leads to less detected corners and vice versa. An alternative definition of the function R is given by Nobel [4] which doesn't need the parameter k:

$$R := D / (T + \epsilon)$$

with

$$\epsilon > 0.$$

Finally the detected corners are grouped, so that in a certain neighbourhood around a corner no other corner with a higher R-value exists. A usual size of this neighbourhood has a radius of 2 to 10 pixels.

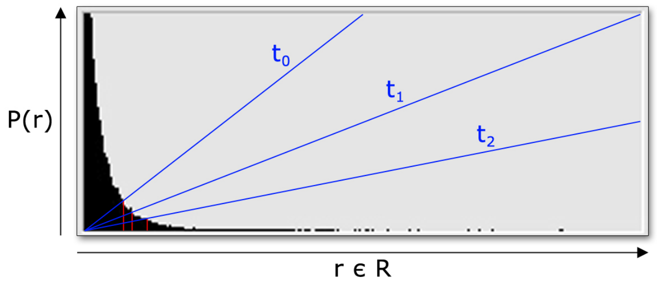


Fig. 1: Characteristic distribution of R-values in an image and derived thresholds t_0, t_1 and t_2.

Although the alternative definition by Nobel reduces the number of parameters by one, a threshold for the R-value has still to be choosen. In a real-world application, especially in the underwater domain, this threshold usually varies greatly depending on the particular scene. For low contrast images, e.g. in turbid water, the threshold should be rather low. Unlike this, in clear deep water with artificial lighting the images have a very high contrast and accordingly the threshold should be rather high. Moreover, in some situations the processed images contain both characteristics, some high contrast objects in the foreground and some low contrast structures in the background. Finding a single suitable threshold in such a situation is often not possible. In addition, for the application described in this paper, a manual selection of a threshold by the ROV pilot would be very inconvenient and not feasible. Therefore, a method to automatically select a good threshold is described in the next section.

B. Automatic Threshold Selection

To determine a good threshold, we use the probability distribution of R-values in the image. Thus, the R-value for every pixel is calculated in a first step and the global probability density function (GPDF) P_G is approximated using these values. Truncating all R-values with $P_G(r) = 0, r \in R$, a characteristic distribution is obtained. The intersection of a line through origin with slope s and the GPDF results in a suitable and contrast-independent threshold for R. Fig. 1 shows an exemplary distribution with thresholds t_0, t_1 and t_2 obtained through different slopes. The slope s is proportional to the relative amount of features which are found.

The threshold obtained by the previously described method works fine as long as each image has a uniform contrast. The method compensates only for contrast changes between images, e.g. for different operational environments, but does not compensate for contrast changes within one image. As aforementioned, situations where large contrast differences within one image exist, are quite common in the underwater domain. Therefore, the described approach has to be extended to be able to handle local contrast differences too:

The image is regularly divided, e.g. 4×4, into subregions and for every subregion the local probability density function (LPDF) P_L is approximated. Calculating a threshold as per description for every subregion would gain some locality, but would also lead to irregular features in subregions which are very uniform and which have practically no structure at all. In those regions, the above method would find a very low threshold such that even the finest noise would result in some features found. To prevent this, it has to be decided for each subregion to which extent the local PDF or the global PDF should be used. As a measure which allows for such a differentiation the Shannon entropy can be used. The entropy is an abstract measure for the information content of a local area and can be derived from the particular LPDF:

$$H_L := -\sum_{r \in R} P_L(r)) \log P_L(r)).$$

Accordingly we can define the global entropy H_G with the global PDF:

$$H_G := -\sum_{r \in R} P(r)) \log P(r)).$$

With these two values, the local entropy H_L of an image region and the global entropy H_G of the overall image, we can now define an entropy-driven probability density function (EPDF) P_E for each subregion which is composed of the local and global PDF:

$$P_E(r) := \begin{cases} P_L(r) & H_L > H_G \\ P_L(r) * \frac{H_L}{H_G} + P_G(r) * \left(1 - \frac{H_L}{H_G}\right) & H_L \leq H_G \end{cases}$$

with

$$r \in R.$$

Using these EPDFs to derive the particular thresholds for every subregion results in a noticeable improvement of feature detection in non-uniform (with respect to their contrast) images. Despite this improvement, this solution may suffer from unstable features when they are tracked over several images and the features *move* from one subregion to another with a different threshold. One possible solution to this problem would be the calculation of separate subregions for each single pixel, but this approach would have relatively high computational costs. A less costly approach is the interpolation of a threshold for each single pixel based on the thresholds of the subregions. As a preparation step for the interpolation, the thresholds of each subregion are propagated to the corners $\{c_0, \ldots, c_3\}$ of each region. The threshold c at a particular corner is the average of the thresholds of all incident subregions to this corner. Let (d_x, d_y) be the position of a pixel d in a subregion and let (sr_w, sr_h) be the dimensions of that subregion. The individual threshold t_d for d is then calculated by:

$$\begin{aligned} t_d := \quad & c_0 * (1.0 - u - v + w) + \\ & c_1 * (u - w) + \\ & c_2 * (v - w) + \\ & c_3 * w \end{aligned}$$

with

$$u := d_x / sr_w$$

$$v := d_y / sr_h$$

$$w := (d_x * d_y) / (sr_w * sr_h).$$

This interpolation ensures a smooth transition between the thresholds of the different subregions while being computationally cheap. Fig. 2 shows a comparison between the results of the Harris detector with the described automatic threshold selection and the results obtained with the "good features to track" algorithm by Kanade/Tomasi (using the openCV implementation) for different underwater scenes. As it can be seen, the number of feature points selected by the automatic thresold stays low and nearly constant. This provides a stable basis for any kind of feature tracking. Contrary, the standard "good features to track" algorithm needs a scene-dependend adjustment of its threshold. This adjustment would have to be done by the pilot of a system during operation and would not be very feasible. In addition, autonomous tasks like mosaicing of large sea floor areas also rely on stable features. These tasks can hardly be done with the classical feature detection algorithm. In this sense, the described approach for automatic threshold selection lays out the basic abilitiy for robust feature detection in real world scenarios and applications. To summarize the described algorithm the schematic operation is depicted in fig. 3 and the algorithm can be itemized as follows:

- Calculate R-values for all pixels.
- Approximate the GPDF.
- Divide the image into subregions.
- Approximate the LPDF for each subregion.
- Calculate global and local entropy values.
- Compose an EPDF based on the ratio between global and local entropy for each subregion.
- Calculate threshold values for each subregion using the EPDF.
- Propagate the threshold values to the corners of the subregions.
- Interpolate an individual threshold for each pixel based on the threshold values at the corners of the subregion the pixel lies within.

With this approach, features can be detected under various lighting and contrast condiditions without further parametrization. This makes it an ideal basis for feature tracking and motion estimation in real world applications.

C. *Feature Tracking*

Once a set of robust features is detected, these features are tracked with an iterative version of Lucas-Kanade optical flow algorithm in pyramids as described by Bouguet in [5]. The key idea of this iterative version of the Lucas-Kanade optical flow algorithm is to use an image pyramid representation to allow for accurate tracking (i.e. small integration window) and robustness against large motions (i.e. large integration window) simultaneously. In addition to the robustness against large motion the algorithm is also quite robust against changes in lighting or contrast. The implementation of the feature tracker that was used is available in the OpenCV library [6].

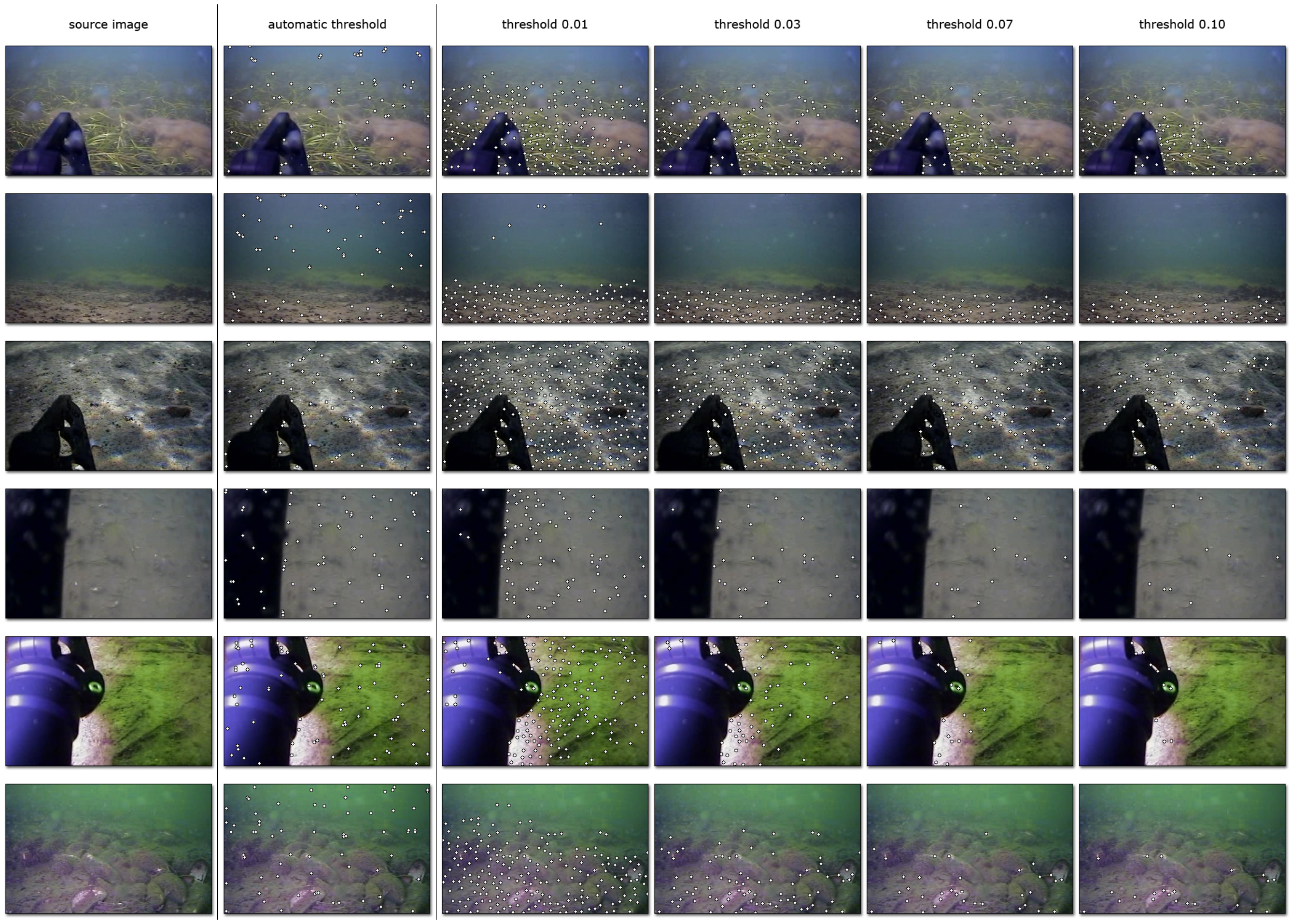


Fig. 2: Comparison of the Harris feature detector with automatic threshold selection and the "good features to track" with several discrete threshold values.

Fig. 4: The used vehicle. A LBV150B2 by Seabotix Inc.

III. Application to a small-class ROV

Using the described vision algorithm to estimate the motion in a live video feed, a video-based "hover control" for a small-class ROV was implemented.

A. *LBV150*

The used vehicle is a LBV150 (fig. 4) by Seabotix Inc. [7]. The main specifications of the vehicle are as follows:

- Dimensions of $530mm \times 245mm \times 254mm$ (l/w/h).
- Weight of 10.4 kg in air.
- Thruster configuration: two forward thrusters, one vertical and one lateral thruster.
- Max. operating current is 1.5 knots.
- 570 line color camera (PAL)

B. *System Setup*

The vehicle is normally controlled by a hand controller which is connected to the surface power supply. The hand controller uses a RS232 signal to communicate with the ROV. The signal is modulated onto the DC power inside the surface power supply and is demodulated onboard the ROV. In order

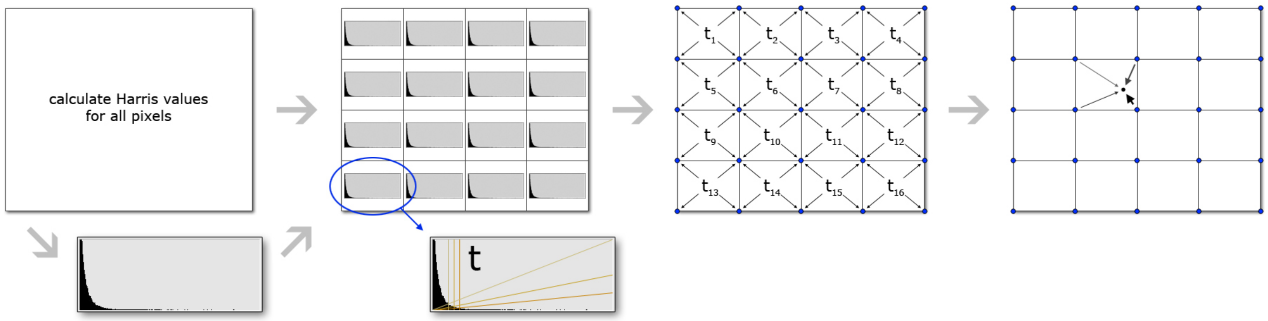
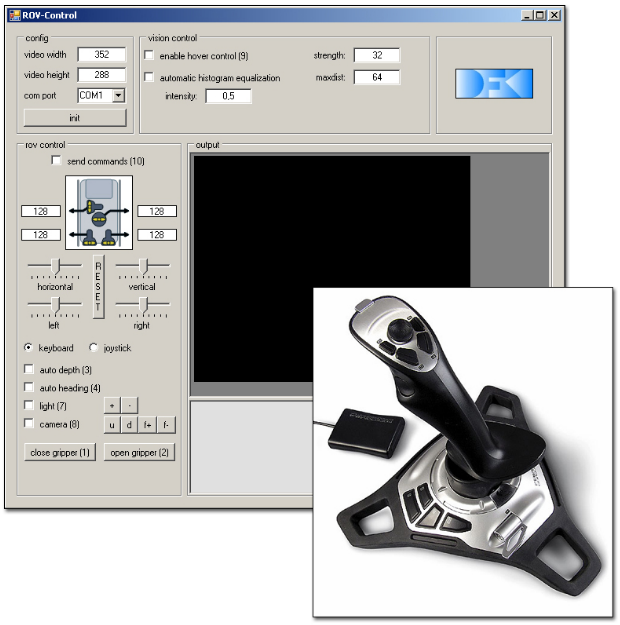


Fig. 3: Schematic operation of the automatic threshold selection.



video width 352
video height 288
com port COM1
[ ] enable hover control (9)
[ ] automatic histogram equalization
intensity: 0,5
strength: 32
maxdist: 64
[ ] send commands (10)
128 128
128 128
[✓ keyboard] joystick
[ ] auto depth (3)
[ ] auto heading (4)
[ ] light (7) + -
[ ] camera (8) u d f+ f-

Fig. 5: Screenshot of the developed control software and the 4DOF joystick used for the computer control of the LBV.

to be able to control the vehicle with a computer, the protocol sent by the hand controller was reimplemented such that the computer could completely replace the manual control. As the hand controller has only a control frequency of just $2Hz$, the computer is also bound to this rather slow control frequency. Attempts to increase the frequency resulted in problems with the video processing onboard the ROV. Given that a higher frequency is clearly beyond the specifications, it is by no means devaluating the Seabotix system. The system is just not originally designed for computer control.

In our setup, the ROV is controlled via a software running on a standard desktop computer or notebook. The software displays information of the actual system state, e.g. the actual thrust on the different thrusters, and has an integrated video window which displays the video feed of the ROV. Additional information is overlayed while the hover control is activated, e.g. the selected features or the vector of the average relative movement. To steer the ROV manually, a 4DOF joystick is connected to the software (fig. 5). In addition to the raw movement control of the vehicle, all other system functions can also be accessed via several joystick buttons. In comparison to the classic hand controller we made the experience that the joystick control of the vehicle is much more convenient.

The hover control can be activated/deactivated at any time with the respective button on the joystick. When the hover control is activated, the system selects a number of salient features around the center of the actual video frame using the previously described vision algorithm. Subsequently this set of features is tracked by the Lucas-Kanade optical flow until the hover control is deactivated again. During the hover control the inputs of the joystick are superimposed on the commands given by the software. As described in the next section in more detail, the overlay of manual commands allows for a change of the relative position in front of the observed structure, e.g. a little thrust on the lateral thruster can cause the ROV to circuit around the object of interest.

C. PID Control

While the hover control is activated, the response of the ROV to the relative movements detected by the vision system can be specified due to five PID controllers (fig. 6). The PID controllers are highly parameterized to be adaptable to different vehicle and thruster types. For example, the forward thrusters on the LBV have a preferential direction and thus the output of the correspondent PID controllers is asymmetric. Otherwise every rotational movement would also result in a forward movement.

The first three PID controllers control the lateral thruster and both forward thrusters and receive the average horizontal movement as input. Depending on the strength of each of these controllers, the vehicle can be set to rotate towards the seen structure, to perform a lateral movement or to perform a movement composed of these two behaviours. The lateral component is useful to compensate for lateral current, whereas the rotational component allows to circuit around the observed structure by controlling only the lateral thruster manually. The task of the fourth PID controller is to regulate the depth of

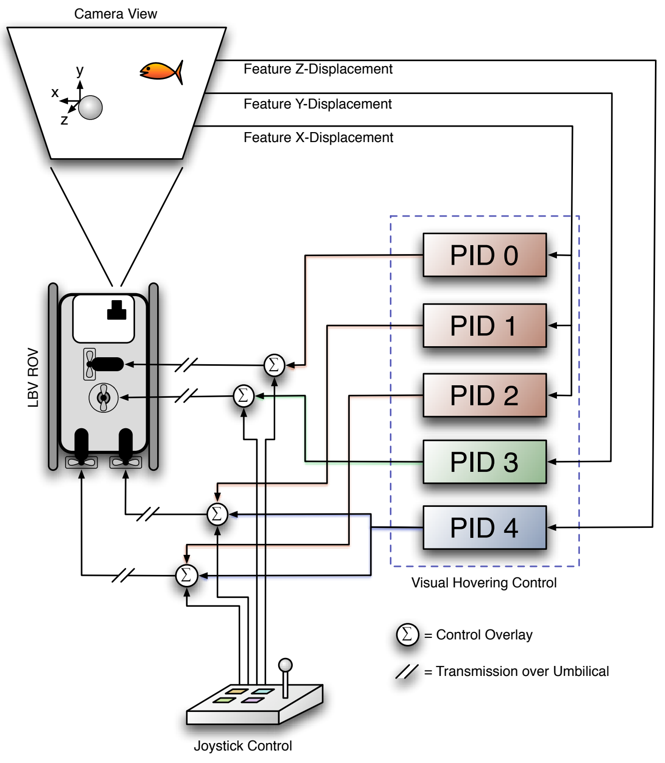


Fig. 6: The hovering control scheme with 5 PID controllers.

the vehicle. It controls the vertical thruster and receives the average vertical movement as input. The fifth PID controls the distance to the observed object by controlling both forward thrusters. The input to this controller is the average distance between all pairs of features in the given feature set. When the inter-feature distance is decreasing, the vehicle's distance to the observed structure is increasing and vice versa.

D. Test Environment

The previously described system was tested in the DFKI underwater testbed. The testbed consists of a $25m^3$ tank spanned by a 3D gantry crane. The gantry crane allows an object to be precisely moved in all 3 dimensions inside the tank with speeds up to 5 meters per second. To test the hover control, the LBV is positioned manually in front of the tip of the gantry crane. As the tip is now in the center of the video image, the features which are selected after the activation of the hover control, usually belong to the tip of the gantry crane. In this situation, the ROV is *locked* onto the tip of the crane (see fig. 7a). As long as the hover control is activated, the ROV will follow the movements of the gantry crane.

With this setup, it is possible to simulate the influence a current would normally have on the ROV, e.g. if the tip of the crane is moved down, the visual appearence is equal to a current dragging the ROV up (see fig 7b). The maximum speed of the tip of the gantry crane in this setup was 0.5 knots. At this speed the ROV had no problems to follow the tip. It turned out that the main difficulty with the control of the vehicle lies within the very low control frequency. At only $2Hz$ it is not possible to control the vehicle at higher speeds without having it to overshoot when fast changes in the movement direction of the gantry crane tip occur. However, in a real world application these very fast changes in the current dragging the vehicle are very unlikely to occur. Thus it can be assumed that the maximum current which can be compensated is considerably higher than 0.5 knots even with a control frequency of only 2Hz. A more extensive test phase in a real enviroment is scheduled within this year.

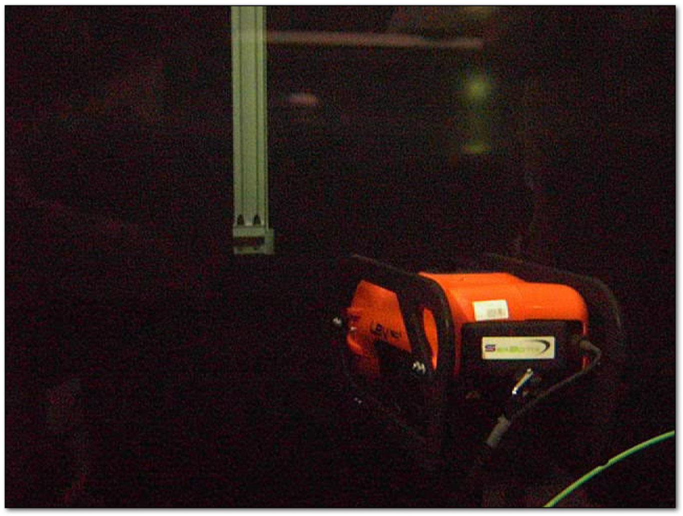
IV. Conclusion and Outlook

We presented a control system which enables off-the-shelf small- and middle-class ROVs to autonomously hover in front of a structure or an other object of interest. The approach uses solely the on-board camera and thus does not need a modification of the ROV itself. The used vision algorithms are aimed at real world applications and are robust enough to handle various light and visibility conditions.

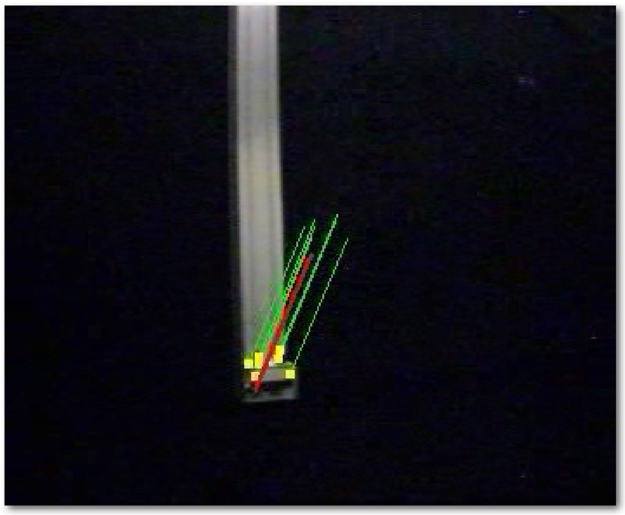
The area of application for the presented technique will be extended to tasks like object following, visual odometrie and mosaicing. The groundwork for these applications lies within the robust and scene-independend feature selection algorithm described in this paper. Furthermore, the integration of additional sensor data, e.g. IMU or Sonar data, is planned and is expected to enhance the overall robustness and precision of the systems.

References

[1] Moravec, H.: Rover Visual Obstacle Avoidance. In: proceedings of the seventh International Joint Conference on Artificial Intelligence, 1981, S. 785-790
[2] Harris, C., Stephens, M.: A combined corner and edge detector. In: proceedings of The Fourth Alvey Vision Conference, 1988, S. 147-151
[3] Tomasi, C., Kanade, T.: Detection and Tracking of Point Features / Carnegie Mellon University. 1991 (CMU-CS-91_132)
[4] Nobel, J.A.: Description of Image Surfaces, Department of Engineering Science, University of Oxford, UK, Diss., 1989
[5] Bouguet, J.Y.: Pyramidal Implementation of the Lucas Kanade Feature Tracker, Intel Corporation Microprocessor Research Labs, 2000
[6] http://sourceforge.net/projects/opencvlibrary/
[7] http://www.seabotix.com/



(a)



(b)

Fig. 7: Fig. 7a shows the LBV hovering inside the DFKI underwater tank *locked* onto the tip of the 3D gantry crane. Fig. 7b depicts the relative movement of the tip of the gantry crane detected by the described vision algorithm. The yellow dots mark the selected features, the green lines mark the individual movement of the features and the red line marks the average movement of all features.