Lights and Camera: Intelligently Controlled Multi-channel Pose Estimation System

Olena Borzenko[1], Wei Xu[1], Mark Obsniuk[1], Arjun Chopra[1]
Piotr Jasiobedzki[2], Michael Jenkin[1], Yves Lespérance[1]
[1]Department of Computer Science and Engineering, York University, Toronto, Canada.
{olena, xuwei, mobsniuk, chopra, jenkin, lesperan}@cs.yorku.ca
[2] MDA Space Missions, Brampton, Canada.

Abstract

Guiding the spacecraft docking process requires the use of sensors that estimate the relative position of the two vessels. This task is complicated by the widely variable on-orbit illumination. To combat this, controllable docking cameras are augmented by computer-controlled illuminants. But how should these illumination and capture parameters be controlled and how should the images obtained under different conditions be combined in order to estimate the relative pose of the vessels? We address these issues in the "Lights and Camera" system. Images captured with the same camera and scene geometry but under different lighting conditions are merged, and the resulting edges are used to estimate the target's pose. A high level controller monitors the imaging process and determines the set of images to capture and use for pose estimation. This paper describes the "Lights and Camera" system architecture and initial results of its operation on mockups of space hardware.

1. Introduction

Rendezvous and docking are key requirements for assembling, retrieving, refueling, and repairing spacecraft, such as satellites, space shuttles, large space platforms, and even space stations (see Figure 1). Although spacecraft rendezvous and docking routinely occurs with humans "in the loop", more autonomous operation is desired. Achieving "soft" docking with the target (the uncontrollable part, such as a docking platform) requires precise control of the relative velocity of the chaser (the vehicle whose movement is controlled). As a result, an essential requirement for autonomous docking is adequate sensing of the relative position and orientation between the docking spacecraft.

Vision systems that support docking craft in space are faced with the highly variable lighting conditions associated with the outer space environment (see [10]). To overcome unfavorable natural illumination, the camera is often accompanied by one or more (typically fixed) light sources that can be controlled, and the camera itself has controllable image capture parameters. The task of the human operator or software agent is to manipulate the light conditions and camera parameters to appropriately illuminate and capture portions of the scene that are critical to the task at hand.

To adequately capture the entire scene in one acquisition is often impossible. What is possible, however, is to capture images under different illumination and camera conditions (see Figure 2) and then to process these images together in a meaningful way. During the final phases of space docking, the spacecraft rate of approach is extremely low. As a result the scene can be treated as being essentially static over a small temporal window. Thus multiple images with the same image geometry, but captured under different lighting conditions are available in order to localize the target precisely. For the multiple images to enhance the information available over a single image, not only must the different illumination and image capture conditions be chosen in some intelligent manner, but also an effective process is required to merge information from these images.

Given that the spacecraft docking system can collect multiple images of the scene with the same geometry, how difficult is the task of determining the appropriate set of images to use to localize the docking target? Suppose that the system is capable of controlling three illuminates each with eight possible illumination settings, and that the image acquisition system has 15

MACDONALD DETTWILER
Space and Advanced Robotics Ltd.
CHASER
TARGET
Docking Speed: 20 cm/s
Misalignments: 5 deg in pitch

Figure 1. Spacecraft docking: a simulation of the overall task (courtesy of MacDonald, Dettwiler and Associates Ltd., www.mdacorporation.com).

possible capture setting (this describes our testbed system, operational systems have even higher degrees of freedom). Taking into account sets of images rather than single images, the search space size for the task is of size $2^{8x8x8x15}-1 = 2^{7680}-1>10^{2311}$. An exhaustive search of the space is impractical and there is clearly a need for an intelligent parameter selection scheme that accumulates and exploits knowledge about which combinations of settings may be beneficial in the current conditions. The development of such a system is the primary goal of the research presented here.

This paper describes the "Lights and Camera" control system that addresses the task of estimating spacecraft pose with controllable illumination and camera parameters. A knowledge-based intelligent controller guides the image acquisition process and selects light configurations. A multi-channel edge detector robustly extracts edge features that are statistically significant over the collection of images of the target, and the final pose estimate is produced from a line-based model-matching genetic algorithm.

The rest of the paper is organized as follows. Section 2 discusses related work on knowledge-based control of vision and robotics. Section 3 describes the architecture of the "Lights and Camera" system and its main image processing and control components. Experimental results and a sample run of the system are presented in Section 4. Section 5 concludes the paper and outlines directions for future work.

2. Related work

A number of different approaches to adaptive and knowledge driven operation of vision systems have been proposed in the literature. Shekhar et al. [20] package basic image processing algorithms into multilayer "smart" modules that encapsulate

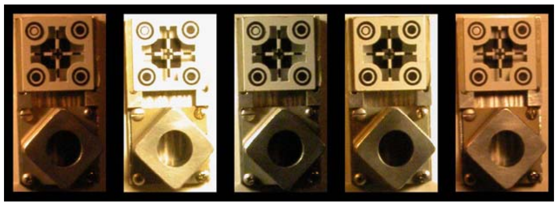


Figure 2. Views of the Dexterous Handling Target and Micro Fixture under various illumination conditions. Pose estimates computed from these images are used to capture the fixture autonomously.

knowledge of how to run the associated algorithm, evaluate its results, and tune its parameters. First, vision tasks are decomposed into a hierarchical plan. Rules expressing possible approaches, constraints, parameter adjustment, and failure handling are then used to complete the plan. Automatically solving a specific problem entails the selection, ordering, and deployment of vision modules, and the tuning of their parameters on training images.

Robertson [17] uses the concept of procedural reflection to allow monitoring and modification of the computational state of an image processing filter. The adaptive operation of the vision system, incorporating the adaptable filters, is comparable to the control of a closed-loop system.

Shanahan [18][19] describes an Event Calculus-based active vision framework that incorporates feedback and expectation, linking low-level sensing and high-level reasoning. In this framework, an agent generates hypotheses about possible explanations of vision sensor data and selects an action – such as tuning a vision parameter or manipulating a robotic arm – that is expected to disambiguate among these hypotheses.

Among recent work on knowledge-driven robotic agents is the development of executive control systems based on Intelligent Distributed Execution Architecture, IDEA. The basic hypothesis of IDEA is that a large control system, such as that for a rover [8] or a rescue robot [4], can be structured as collection of interacting agents, each with the same fundamental structure, implemented for example in an agent language such as the concurrent temporal version of Golog [16].

The Intelligent Controller we use in the "Lights and Camera" system is an extension of the controller described in [1] and is implemented using the high-level model-based agent programming language *IndiGolog* [6][13], an extension of ConGolog [5] and Golog [14]. In IndiGolog, the programmer provides a declarative specification of the domain – actions and their preconditions and effects on fluents, i.e. the dynamic aspects of the state − and develops complex control programs in terms of these. IndiGolog supports both planning and high-level reactivity, in contrast to classical planners on the one hand and purely reactive architectures on the other. Agents written in IndiGolog are capable of sensing the (changing) environment and communicating with other agents or software modules, as opposed to expert system shells. The IndiGolog interpreter that we use in the project is implemented in Quintus Prolog and SWI Prolog.

For the space docking application, the arguments in favour of using IndiGolog are mainly related to software engineering. For an agent to be able to reason about the perceived world and the vision system, it must keep track of all of the relevant aspects (states) of the environment and of the vision modules. Programming languages such as C, C++, or Java do not require the designer to specify system states and their dynamics explicitly. As a result, they are not particularly suitable for the design of such intelligent controllers. In contrast, IndiGolog provides a transparent and scalable framework that encourages declarative description of a domain and automatically reasons about state updates. In addition, it supports the agent's ability to interact with other modules and sense the environment.

3. The Lights and Camera system

The Lights and Camera system consists of the three primary components: the Image Server, the Vision Server, and the high-level Intelligent Controller (see Figure 3). The Image Server operates a digital camera and associated lights, providing control over parameters such as light intensities, camera aperture, shutter speed, and focal length. The Vision Server encapsulates the image processing pipeline consisting of multi-channel edge detection, edge linking, and model matching for pose estimation. The Intelligent Controller manages the image acquisition process. (The individual stages are described in detail below.)

As long as the static scene assumption holds, the following feedback loop operates. Following a request from the Controller to capture a new image under a specific set of light and camera parameters, the Vision Server queries the Image Server for the corresponding image. The new image is captured and added to the existing collection of images taken under different illumination conditions and camera parameters. The Vision Server updates the composite edge map based on the current image set and estimates the pose of the target. The uncertainty of the current result, measured using the RMS error in the model matching stage, is then sent to the Controller, closing the feedback loop.

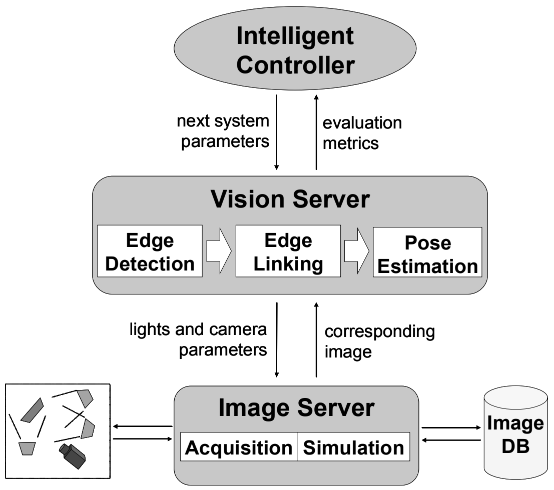


Figure 3. The Lights and Camera system architecture.

The Controller module then determines whether the current image should be kept or discarded from the image set and generates a new set of image capture parameters for the next iteration. The Controller terminates this process when the target's pose estimate error falls below a predefined threshold. At any time during the operation, the system maintains its current best estimate of the target's position.

3.1. The vision server

This section describes the three components of the vision processing pipeline; edge detection, edge linking, and model-based pose estimation.

3.1.1. Multi-channel edge detection

The multi-channel edge detection algorithm [21] is an extension of the single-channel Canny edge detector [3] to operate on multiple input images (channels). Figure 4 provides a sketch of the approach. Input images are first processed in separate channels (one image per channel) to obtain individual gradient maps. In a local neighborhood centered at each pixel position, the local gradient orientation samples (weighted by the corresponding gradient magnitudes) are modeled as a two-component mixture of Gaussian's in which the inliers (the normal gradient samples corresponding to the local edge structure) are modeled by the main Gaussian distribution and the outliers (gradients corresponding to shadow edges and other random noise) is modeled by the background Gaussian distribution. A scheme based on Expectation Maximization (EM) algorithm [7] is used to decompose the mixture model, and to identify and

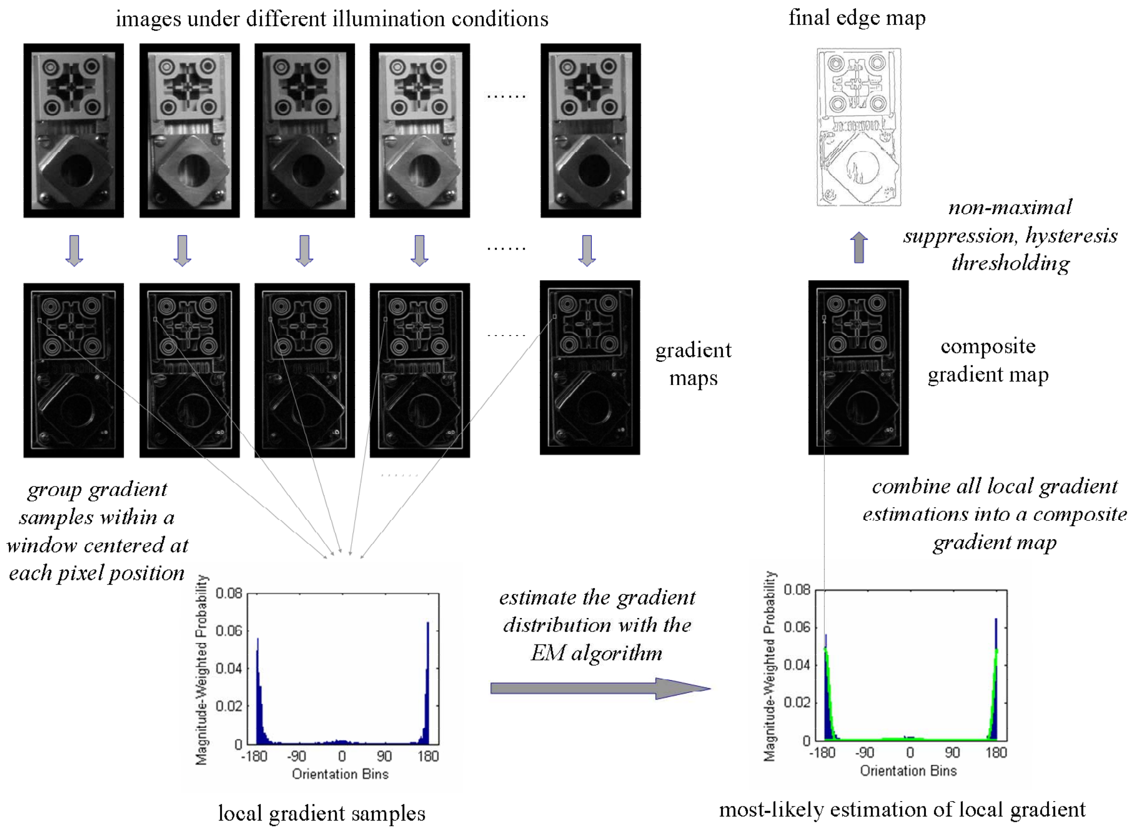

images under different illumination conditions
final edge map
non-maximal suppression, hysteresis thresholding
gradient maps
composite gradient map
group gradient samples within a window centered at each pixel position
combine all local gradient estimations into a composite gradient map
estimate the gradient distribution with the EM algorithm
Magnitude-Weighted Probability
Orientation Bins
local gradient samples
most-likely estimation of local gradient

Figure 4. Outline of the multi-channel edge detection algorithm.

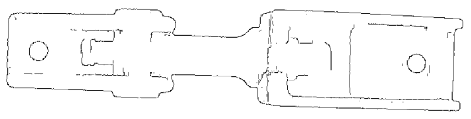
separate the outliers from the inliers. The gradient of the local edge structure is estimated from the distribution of the inliers: its orientation is the mean of the main Gaussian distribution, and its magnitude is defined as the proportion of all the sample magnitudes that are associated with the main distribution. A composite gradient map corresponding to the underlying edge structure is then constructed from the merged local gradient estimates. This "edgel map" is computed from the integrated gradient map using the post-processing techniques of the Canny edge detector [3].

3.1.2. Edge linking

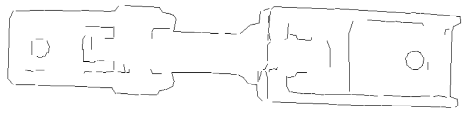
An edge linking algorithm based on [12] is used to link the multi-channel edgels and group them into edge lists. A top-down polyline splitting algorithm [15] is then used to recursively fit line segments to each edge list. The algorithm takes as input the linked edgels. The line segment that approximates the edge list and joins the first and last edge points (x_1,y_1) and (x_2,y_2) is given by $x(y_1-y_2)+y(x_2-x_1)+y_2x_1-y_1x_2=0$. The distance of each edgel in the edge list from this line segment is computed, and this is used to compute the normalized maximum error e for the set of edgels: $e=max_i|d_i|/D$, where D is the distance between the end points of the line segment and d_i is the distance between the line segments and the i-th edgel in the edge list. This error is used to estimate the goodness of fit of the line segment to the edgel list. If the maximum error exceeds a pre-defined threshold value, a new vertex is inserted at the point in the list that is farthest away from the line, and the line segment is split into two new segments. This algorithm is applied recursively to each of these line segments, terminating when the normalized maximum error for all edge points in the list falls below a specified threshold value. After all edge lists have been fitted with segments, the edge linking module creates a list of edges that approximate the original edgel map (Figure 5).

3.1.3. Pose estimation

A genetic algorithm [9] is used to handle the feature correspondence and pose determination simultaneously. The approach is similar to that undertaken by Kawaguchi [11] in that line features are



(a)



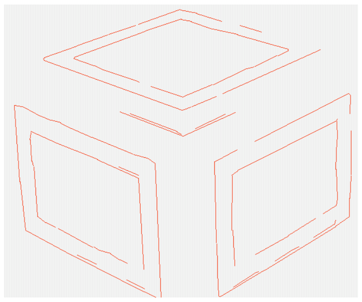
(b)

Figure 5. (a) Multi-channel edge map of a Hubble Space Telescope Latch. (b) Its polyline approximation.

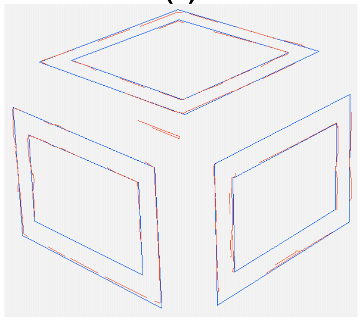
used to match the model to the image. Here the 3D model is back-projected into the view space using a perspective projection. The fitness function incorporates angle and line end-point distance differences. Our approach determines the pose of an object from partial line features taken from an image.

Each of the six pose parameters (R_x, R_y, R_z, T_x, T_y, T_z) is encoded in 8 bits. The range of values for each parameter is based upon an initial seed pose. The rotation parameters are ±25 degree's and translation parameters are ±40 pixels. The initial population is seeded from a random uniform distribution over the entire solution set. For each generation, the fitness of each element of the population is computed and this is used to determine the members of the next generation. The fitness function is the sum of two metrics; the angle difference between model lines and image lines squared and the distance of image line end points and model lines squared. The best-fit image line is matched to each model line.The top five percent of the individuals are passed onto the next generation unchanged. The remaining members of the next generation are generated from the top 30 percent elements of the current generation. Two individuals are randomly chosen from the top 30 percent to create a new individual. A crossover point is chosen and material from the two parents is used to form this new individual by choosing one portion from one parent and the remainder from the second parent. One bit of this individual is then mutated. If the resulting individual is unique, then it is added to the population. The genetic algorithm operates for a fixed number of generations. The output from the algorithm is the solution corresponding to the individual with the best fitness.

The algorithm typically converges to a stable set of values within 10 generations with a population size of



(a)



(b)

Figure 6. (a) Multi-channel polyline data for a textured cube. (b) The resulting model match data overlaid on (a).

200 individuals. The computational cost increases with the number of generations and larger population sizes. A number of trials were conducted with varying populations sizes and number of generations. The value of 200 individuals was chosen as a compromise between accuracy and speed. Figure 6 shows an input polyline representation and the resulting model-polyline match.

3.2. High-level Controller

To obtain a precise pose of the target, the high-level controller manages the image acquisition process and determines the set of input images for the multi-channel vision processing pipeline. This optimization task is complicated by the size of the search space and the complexity of the effects that changes in system parameters produce on the pose estimate.

The Intelligent Controller maintains qualitative representations of system states, e.g. “low lighting coming evenly from all the three light sources”, in essence discretizing along the “light brightness” and “light directionality” dimensions. These qualitative representations permit reasoning about the similarity of image acquisition states and, consequently, the similarity of images taken under corresponding states. Moreover, this allows human domain knowledge to be expressed in a natural way and reduces the search effort required for image capture settings. Based on the

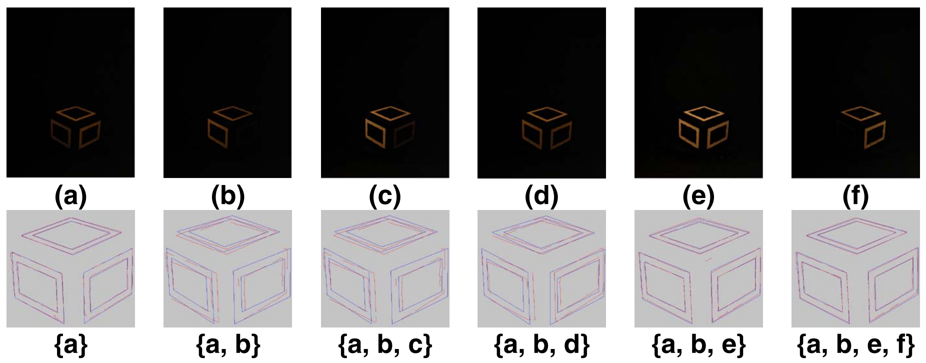


Figure 7. The images acquired (top row) and the model matching result and edges recovered (bottom row) during the six iterations of the sample run. Images (a), (b), (e) and (f) were selected by the Controller for the final pose estimation.

described classification of images and system states, the Controller employs a greedy heuristic search for next system parameters, based on the likelihood of a set of images to share common information (see [1][2] for more details).

The high-level control procedure for image acquisition and merging is as follows. At the first step of the algorithm, the settings for the initial image are picked randomly. The agent then iteratively chooses light intensity settings for the next image acquisition, seeking images that are expected to contribute additional information with respect to the existing image set and merit function.

If the merit function does not increase sufficiently due to the newly added image, the image is marked to be discarded, the parameters under which it was taken are marked as unfavorable, and the search for a new addition to the set is initiated. When searching for the best additional image, the controller restricts its choice to the images that are not similar either to the current set of images or to the images discarded at earlier steps. Augmenting the composite continues until a maximum number of iterations have been performed or the merit function reaches a predetermined level.

4. Sample run and experimental results

Figures 7 and 8 illustrate a sample run of the Lights and Camera system for a cubical mockup target. The initial image capture parameters are chosen randomly and an initial estimate is calculated. At each iteration, the Controller obtains a new image, evaluates the effect of its addition on the quality of pose estimate, and thus determines if this image should be included in the set of images for future processing. In this example, after six iterations the match error measure falls below the preset threshold and the system outputs the final pose of the target. Four images were combined to produce this final estimate. Figure 8 shows the dynamics of the size of the current image set maintained by the system as well as the quality of match produced by the pose estimation module for each of the image sets.

Running times excluding the image capture time averaged 1480 seconds for the current control system. Additional experimentation was carried out with the same cubical target to compare the performance of our heuristic control approach and basic control approaches. The results in terms of average running times over 5 runs are given in Figure 9. Here the "simple" controller acquires and merges images under randomly selected settings. The "greedy" controller acquires and merges images in a similar fashion, then evaluates system performance, and discards the last image from the image set if the error does not decrease. The "heuristic" controller performs a greedy search for "not similar" settings and is the one described earlier in the paper. The same error threshold was used for all three controller types. As seen from Figure 9, discarding images that do not provide an immediate increase of the quality of match improves convergence, as does using a heuristic search for image acquisition settings.

5. Conclusions and future work

Complex vision tasks – such as automating spacecraft docking – require solutions to a wide variety of image processing problems, from camera calibration to edge detection to high level system control. In the Lights and Camera system, the high level knowledge-based controller, implemented in the model-based agent programming language IndiGolog, controls image acquisition and multiple image processing in the system in order to best estimate the relative pose of the camera with respect to a modeled object.

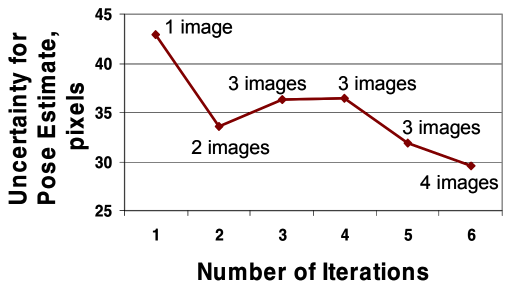


Figure 8. Sample run of the "Lights and Camera" system. For each iteration of the control loop, the number of images used in processing is given.

The system has been tested on a number of targets in a laboratory-based setting. In the future we plan to test the system in a more realistic space simulation on earth and to incorporate the pose estimates obtained by the system within a complete docking control system.

Ongoing developments include a system configuration that provides a line-by-line evaluation of the performance of pose estimation stage. Based on this detailed error metric, the high-level controller will employ more precise, shape-dependent image acquisition and processing strategies.

The process of computing pose from the set of available input images is quite complex involving a number of different phases from multi-channel edgel detection, through edge linking and genetic algorithm-based model matching. Each of these modules incorporates tunable parameters. Planned future work includes incorporating parameter tuning mechanisms within the controller to provide intelligent selection and control of the various parameters that are available.

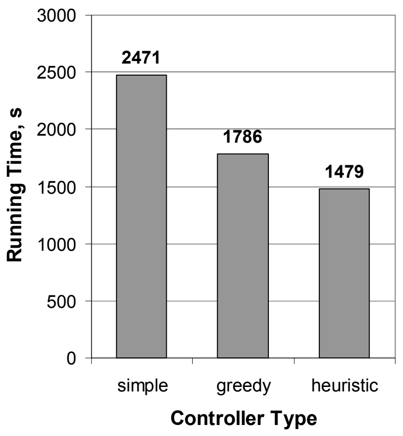


Figure 9. Comparison of system running times for basic controller types.

References

[1] Borzenko, O., Lespérance, Y., Jenkin, M., Controlling Camera and Lights for Intelligent Image Acquisition and Merging, Proc. 2nd Canadian Conference on Computer and Robot Vision, Victoria, BC, pp. 602-609, 2005.

[2] Borzenko, O., Knowledge-Based Control of Vision Systems: Sample Controllers and Design Tools. M.Sc. Thesis (in preparation), York University, Toronto, 2005.

[3] Canny, J.F., A Computation Approach to Edge Detection, *IEEE PAMI*, 8: 679–698, 1986.

[4] Carbone, A., Finzi, A., Orlandini, A., Pirri, F., Ugazio, G., Situation-Aware Rescue Robots, Proc. 2005 IEEE Workshop on Safety, Security and Rescue Robotics, Kobe, Japan, pp. 182-188, 2005.

[5] De Giacomo, G., Lespérance, Y., and Levesque, H.J., ConGolog, a concurrent programming language based on the situation calculus, *Artificial Intelligence*, 121: 109-169, 2000.

[6] De Giacomo, G., and Levesque, H., An incremental interpreter for high-level programs with sensing. In H. J. Levesque, and F. Pirri, (Eds.) *Logical Foundations for Cognitive Agents: Contributions in Honour of Ray Reiter*, Springer, Berlin, pp. 86-102, 1999.

[7] Dempster A.P., Laird N.M., Rubin D.B., Maximum likelihood from incomplete data via the EM algorithm, *J. Roy. Sta. Soc.*, B39: 1-38, 1977.

[8] Finzi, A., Ingrand, F., Muscettola, N., Model-based executive control through reactive planning for autonomous rovers, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2004, vol. 1, pp. 879-884, 2004.

[9] Holland, J. H., Genetic algorithms and the optimal allocation of trials, *SIAM J. Comput.*, 2(2): 88–105, 1973.

[10] Inaba, N.; Oda, M., Autonomous satellite capture by a space robot: world first on-orbit experiment on a Japanese robot satellite ETS-VII, Proc. Int. Conf. Robotics and Automation, Vol. 2, San Francisco, 2000, pp. 1169-1174.

[11] Kawaguchi, T., Nagata, R., Sinozaki, T., Detection of Target Models in 2D Images by Line-Based Matching and a Genetic Algorithm, Proc. Int. Conf. Image Processing, ICIP 99, pp. 710–714, 1999.

[12] Kovesi, P. D., *MATLAB functions for computer vision and image analysis*. School of Computer Science & Software Engineering. The University of Western Australia.

[13] Lespérance, Y., Ng, H.-K., Integrating Planning into Reactive High-Level Robot Programs, Proc. Second Int. Cognitive Robotics Workshop, Berlin, pp. 49-54, 2000.

[14] Levesque, H. J., Reiter, R., Lespérance, Y., Lin, F., and Scherl, R. B., GOLOG: A Logic Programming Language for Dynamic Domains, *J. Logic Program.*, 31: 59-84, 1997.

[15] Jain, R., Kasturi, R., Schnuck, B. G., *Machine Vision*, McGraw-Hill, Inc., 1995.

[16] Reiter, R., *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems,* MIT Press, 2001.

[17] Robertson, P, Brady, M., Adaptive Image Analysis for Aerial Surveillance, *IEEE Intelligent Systems*, 30-36, 1999.

[18] Shanahan, M., A Logical Account of Perception Incorporating Feedback and Expectation, Proc. KR 2002, Toulouse, France, pp. 3-13, 2002.

[19] Shanahan, M., Randell, D., A Logic-Based Formulation of Active Visual Perception, Proc. KR 2004, Whistler, Canada, pp. 64-72, 2004.

[20] Shekhar, C., Moisan, S., Vincent, R., Burlina, P., and Chellapa, R., Knowledge-based control of vision systems, *IVC,* 17: 667-683, 1999.

[21] Xu, W., A Multi-channel Approach to Edge Detection, M.Sc. Thesis. York University, Toronto, 2005.