Spatial Knowledge Processing within the Reproductive Perception Paradigm

Andreas Birk
International University Bremen (IUB), Campus Ring 1, 28759 Bremen
a.birk@iu-bremen.de, http://robotics.iu-bremen.de

Abstract

Reproductive perception is a novel approach to perception. It is based on the assumption that perception is predominantly a *generative process*, i.e., that models that represent hypotheses about the current state of the environment generate so-called pseudo-sensor data, which is matched against the actual sensor data to determine and improve the correctness of these hypotheses. This is in contrast to the view that perception is mainly a reductive process where large amounts of sensor data are processed until a compact representation is achieved. Several successful examples of using this approach for spatial world-modeling are presented here. The first one deals with a robot arm and a camera to learn eye hand coordination. In the second example, the successful detection and 3D localization of humans in single 2D thermal images is described. In the last but not least example, work in progress on the generation and usage of compact 3D models of unstructured environments on a mobile robot is presented. The reproductive perception approach in all three examples does not only influence the way the spatial knowledge is generated and represented, but also its usage especially with respect to the classification and recognition of objects.

Introduction

Computer science approaches to perception are dominated by the view that perception is a process that takes large amounts of data from physical sensors like the pixel array of a camera and feed this data through various stages of processing that each lead to a reduction of the data. This holds especially for computer vision (Zhao *et al.* 2003; Yang, Kriegman, & Ahuja 2002; Brown 1992; Vernon 1991; Horn 1986) but also with respect to more explicitly spatially oriented topics like map building (Thrun 2002). The main idea of so-called reproductive perception is to do the opposite (figure 2). Perception is seen as a process, respectively a series of processes where based on a small model large amounts of data are generated that match the incoming data from the sensors. The processes involved in perception so to say try to reproduce the data delivered by the sensors.

A (world-)model as a compact representation of the environment that is first generated and later on updated by per-

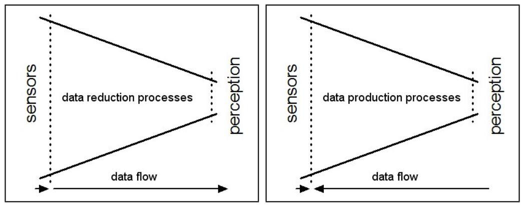


Figure 1: Common computer science approaches to perception process sensor data in various stages that lead to a data reduction (left). The main idea of reproductive perception is in contrast that perception involves processes that generate data that matches the huge amounts of data delivered by the sensors (right).

ception is hence somewhat special within this paradigm. It is not a collection of passively descriptive data, but it can be thought of as a code in a kind of programming language that actually generates data. In the example of learning eye hand coordination presented later on, the objects as well as their spatial properties are for example represented as turtle graphics programs. When executed, these programs produce a collection of data, i.e., they paint a picture in this case, that reproduces the raw data delivered from the sensors, i.e., a camera image in this example.

So, a model is not constructed by stepwise processing of sensor data, but it itself generates large amounts of so-called pseudo sensor data, which is matched against the current sensor data. This generation of the pseudo-sensor data is denoted as rendering. By measuring the similarity between the actual sensor data and the pseudo sensor data, models can be generated and adapted.

This basic idea of reproductive perception is not tied to specific mechanisms. Nevertheless, a concrete implementation is presented here in form of an evolutionary learning scheme (figure 2). As mentioned, a model as the representation of the environment can be thought of as code in a particular programming language. Models are hence programs that can be executed and that generate data. This data is then compared by a metric to the data from the sensors. This metric measures the similarity between the internally generated data and the data from the sensors. This measure of similarity can then be used as a fitness function in an evolutionary

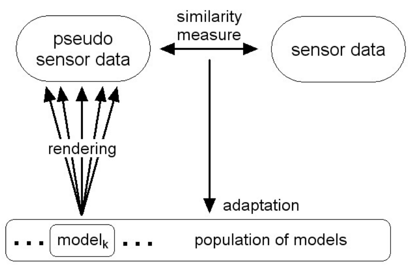
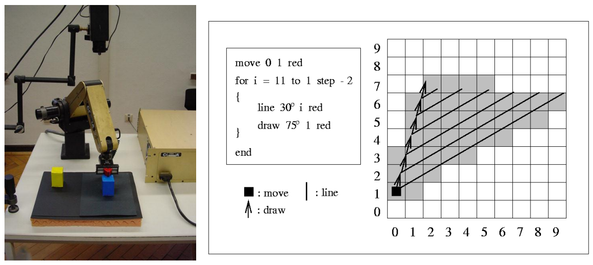


Figure 2: The implementations of reproductive perception presented here use evolutionary learning. A population of evolving models is used to learn a representation that generates so-called pseudo sensor data, which should be as close as possible to the actual sensor data.

algorithm.

Learning of Eye Hand Coordination



```
move 0 1 red
for i = 11 to 1 step - 2
{
     line 30° i red
     draw 75° 1 red
}
end
```

Figure 3: A turtle graphics program re-producing the image of a robot gripper in an experiment where a 3D world-model for eye-hand coordination is learned from scratch.

The application of reproductive perception in experiments with a set-up consisting of a robot arm and a camera learning a 3D world-model related to eye hand coordination (Birk & J. Paul 2000; Birk 1998; 1996) is presented in this section.

The reproductive perception idea is used to generate information about objects, namely the 'body'-parts of the robot-arm as well as colored construction blocks, and their locations in the world in form of programs that generate the 'look' of the objects in the real world, i.e., the data from the camera. This idea builds on (J. Paul & Solomonoff 1995), where theories are viewed as programs which reproduce data from the real world. In the spirit of Occam's Razor and Kolmogoroff, a theory is considered nontrivial if it is 'simpler' than the data it describes, i.e., when it leads to a representation which is shorter according to a complexity measure like the number of occupied memory units.

In the most challenging set of experiments, the representation of the objects in the world, namely the robot-arm and colored building-blocks, was based on a turtle-graphics like language (figure 3). The system had to learn from scratch representations of the objects, i.e., programs that generate images that reproduce the camera data of a scene. This reproductive perception of the sensor data in contrast to a passive processing has a significant advantage as illustrated in figure 4. The system can 'know' about invisible parts of the world. As soon as programs for a building block and the gripper are learned, the system can combine them in a manner where they can represent a scene including occlusions, which are known to cause severe problems to other more common approaches (Kohlhepp *et al.* 2004; Nüchter, Surmann, & Hertzberg 2004).

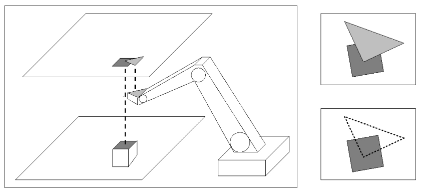
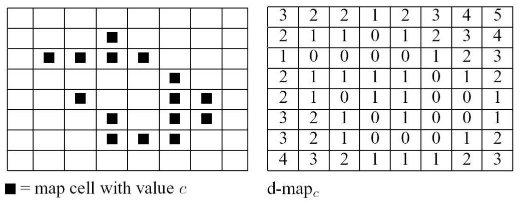


Figure 4: In the reproductive perception paradigm, programs are learned that reproduce sensor data. This has the tremendous advantage that the system can 'know' about invisible parts of the world. If for example turtle graphics programs for a building block and the gripper are learned, the system can combine them to represent a scene including occlusions. For most standard approaches, scenes with occlusions imply quite some difficulties.

The pieces of code that represent the different objects and their locations are learned in an evolutionary process. The programs representing objects are stored in a graph that contains information about motor activations, which are also learned by the system. It is hence possible for the system to link different states of the models, e.g., a program representing 'red robot gripper at location (x, y, z)' and a version of this program representing 'red robot gripper at location $(\hat{x}, \hat{y}, \hat{z})$', by a sequence of motor activations. As a consequence, it can generate plans, e.g., to move its gripper to a particular position. Note that the use of Cartesian coordinates in the above example was only done for illustration purposes. The system represents all objects by programs drawing pictures that reproduce the 'look'. Locations are hence represented in the parameters of the programs, influencing the position of the drawn figure as well as shape properties, e.g., to compensate lens distortions. The experiments with a real world set-up showed that the system was indeed capable of learning a 3D representation of its world in real-time that in the end enable it to manipulate building blocks. More detailed descriptions can be found in (Birk & J. Paul 2000; Birk 1998; 1996).

One crucial aspect for the success of this evolutionary approach is a special metric introduced in (Birk 1996) to compare the similarity of the sensor input with a candidate model. Unlike other approaches using Hausdorff distance (Tung & King 2000; Mount, Netanyahu, & Moigne 1998; Rucklidge 1997) or detecting correspondences between invariant features (Johnson 2000) and using transformations metrics on them (Kohlhepp & Fischer 2001; Basri & Weinshall April 1996), this similarity function ψ has several sig-



3	2	2	1	2	3	4	5
2	1	1	0	1	2	3	4
1	0	0	0	0	1	2	3
2	1	1	1	1	0	1	2
2	1	0	1	1	0	0	1
3	2	1	0	1	0	0	1
3	2	1	0	0	0	1	2
4	3	2	1	1	1	2	3

Figure 5: A distance-map *d-map*$_c$.

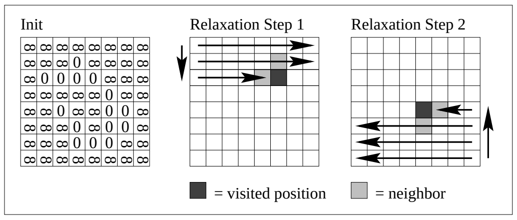


Figure 6: The working principle for computing *d-map*$_c$.

nificant advantages. First of all, it can be computed very efficiently. Second, it is not restricted to particular objects like polygons. Third, it operates on the level of raw data points. It hence does not need any preprocessing stages for feature extraction, which cost additional computation time and which are extremely vulnerable to noise and occlusions.

Concretely, the similarity ψ of two 2D arrays m_1 and m_2 is defined as follows:

$$\psi(m_1, m_2) = \sum_{c \in \mathcal{C}} d(m_1, m_2, c) + d(m_2, m_1, c)$$

$$d(m_1, m_2, c) = \frac{\sum_{m_1[p_1]=c} \min\{md(p_1, p_2) | m_2[p_2] = c\}}{\#_c(m_1)}$$

where

- $\mathcal{C}$ denotes the set of values assumed by m_1 or m_2,
- $m_1[p]$ denotes the value c of array m_1 at position $p = (x, y)$,
- $md(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2|$ is the Manhattan-distance between points p_1 and p_2,
- $\#_c(m_1) = \#\{p_1 | m_1[p_1] = c\}$ is the number of cells in m_1 with value c.

As mentioned, this function can be computed very efficiently, namely in linear time. The algorithm is based on a so called distance-map *d-map*$_c$ for a value c. The distance-map is an array of the Manhattan-distances to the nearest point with value c in map m_2 for all positions $p_1 = (x_1, y_1)$:

$$\text{d-map}_c[x_1][y_1] = \min\{md(p_1, p_2) | m_2[p_2] = c\}$$

The distance-map *d-map*$_c$ for a value c is used as lookup-table for the computation of the sum over all cells in m_1 with value c. Figure 5 shows an example of a distance-map. It can be computed by a relaxation algorithm. The underlying principle is illustrated in figure 6.

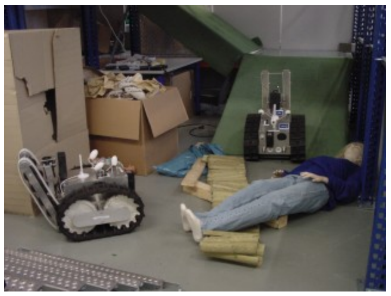


Figure 7: Two of the latest IUB rescue robots in a test scenario. The robots are normally tele-operated but they are also capable of full autonomy including the detection and localization of victims.

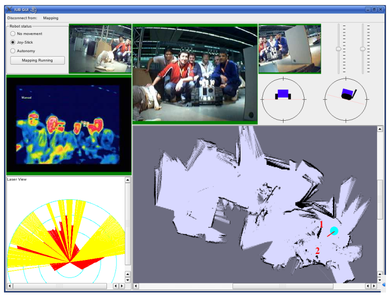


Figure 8: The standard GUI of an IUB rescue robot. The data from the thermal imager can be seen in the left middle window.

Detecting and Locating Humans

The following example of using reproductive perception for spatial processing is based on work in the domain of rescue robotics (Birk & Carpin 2006b), for which the IUB robotics group has developed several systems ranging from the low-level mechatronics (figure 7) (Birk *et al.* 2006b) to high level software enabling full autonomy (Birk *et al.* 2006a) and co-operation (Birk & Carpin 2006a; Rooker & Birk 2006). It is of obvious interest in this domain to automatically recognize victims.

Autonomous human detection is an integral part of many AI applications that provide human-robot interactions, ranging from rescue robotics and security systems to robots serving as guides in museums. While having such a wide application area it is at the same time a very complex task due to the high intra-class variability of human beings – not only do people look very differently among themselves, but also the same person can have very different appearances based on pose, clothing, environment conditions (Mohan, Papageorgiou, & Poggio 2001). Current human detection algorithms fall into one of three categories – *model-based* methods in which a general model is trying to be matched to different parts of an image in order to find a fit (Yuille 1991), *image-invariance* systems which base their matching on a set of image pattern relationships (e.g. brightness levels) that uniquely determine the object being sought (Sinha 1994) and finally *example-based* algorithms that learn detection by being trained on a set of positive and negative examples (Papageorgiou & Poggio 2000; Mohan, Papageorgiou, & Poggio 2001; Oren *et al.* 1997; Yow & Cipolla 1997). In addition these techniques most commonly employ differential imaging in order to detect the silhouettes of human beings and color analysis in order to determine different parts of the human body (Wren *et al.* 1997; Mohan, Papageorgiou, & Poggio 2001; Hogg 1983).

Due to the high complexity of the problem the above algorithms impose restrictions on the environment in which they will be detecting humans or on the to-be-detected humans themselves. Example of such constraints are – greater dynamics of people relative to the background (Wren *et al.* 1997), only one person in the view of the camera (Papageorgiou & Poggio 2000), restrictions on lightning dynamics (Wren *et al.* 1997), people's poses and occlusions (Papageorgiou & Poggio 2000). These assumptions limit the applicability of the algorithms, e.g. in rescue robotics, where people that have to be detected can exhibit a number of these restrictions.

Here work is presented that tackles these problems in two ways. First of all, thermal images are used to ease segmentation. Instead of using a visible light camera, a device with a far infrared image sensor is employed where pixels represent temperatures (figure 8). The second point, which is much more crucial from the application as well as the scientific viewpoint, is to use reproductive perception where a complete 3D scene model is learned on the fly to represent a 2D snapshot. Again, programs are used, which generate images that describe the perceived environment. Here, the environment is modeled as a collection of 3D humans and boxes distributed in space that are projected to a 2D image using OpenGL. The parameters of the OpenGL camera model are roughly based on the parameters of the real camera on a IUB RugBot (figure 7). A human is represented as a composition of its basic body parts – head, torso, arms, legs. The developed human model has 14 rotational joints through which it can mimic almost any pose of a real human. In addition it has six more degrees of freedom (DOF) for the position and orientation of the torso in space. Figure 9 displays the output of a sample program for the case of drawing a whole human and for only an arm. Boxes as the only other components of the 3D scenes are simply defined by their dimensions and positions in space.

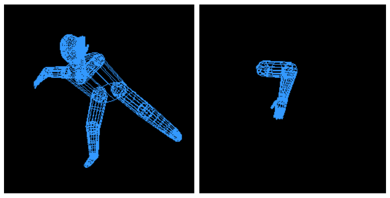


Figure 9: The 2D rendering of a 3D model of a human and an arm drawn in wireframe mode.

Based on the OpenGL routines for drawing humans and boxes a complete drawing program is created as a set of calls to these functions. Each call can be defined as an instruction. An instruction places the corresponding model on a

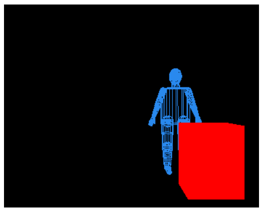


Figure 10: The output of a sample 3D drawing program. The human is drawn in wireframe mode for illustration purposes. When matching renderings to the thermo-images a uniform color texture is used for humans. Note that most of the time, they boxes have a dark color like the background, i.e., they are at room temperature. These dark boxes are mainly used to represent occlusions.

3D scene and after all calls are executed the projection of the drawn scene is taken and returned as the output of the program, as shown in figure 10. A simple thresholding operation is used to reduce the camera image to a bitmap with two colors: the background temperature illustrated by black and temperatures in the human range indicated by blue. Humans emit about 36^{o}C only at exposed body parts. Parts covered by clothes, dust, and so can appear colder. Hence a range of 28^{o}C to 38^{o}C is segmented to be potentially human. Anything else is considered as background temperature.

Figure 11 shows the result from applying this segmentation to an input image.

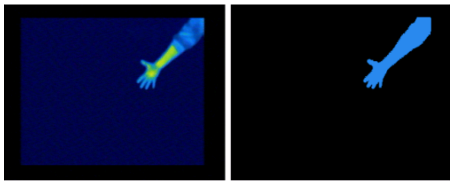


Figure 11: On the left is the original bitmap from the thermo camera. On the right is the same image after segmenting everything in the range of 28^{o}C to 38^{o}C to be potentially human.

The thresholded camera image is then compared to the output of the OpenGL program. For this purpose the image distance function (Birk 1996) shortly introduced in the previous section is also used here. The fitness of an individual in the population is defined via the image distance between the output of the individual and the pre-processed infrared image. In addition to standard evolutionary operators, a hill-climbing operator is used to exploit the meaningful gradients of the image similarity function with respect to basic transformations like translation, rotation and expansion.

The approach has been successfully used in real world scenarios (Birk *et al.* 2006a). Here a few example results are presented. Object recognition, here the identification of humans, is done as follows within the reproductive perception paradigm. The online evolution to generate a model for the current sensor data is bound in time. If the fitness of the best individual in the population is in the end very high, i.e., the similarity between the data it produces and the sensor data is very high, the model can be considered to be correct. Note that this is always the case for all experiments described in this and in the previous section. But it is an additional safeguard to prevent false classifications. The building blocks of this model hence are related to the objects in the real environment. In the example of the victim identification, the best model for the sensor data only contains a 3D OpenGL human model if and only if there is also a human in the related 2D thermal image. This also holds with respect to any number of humans, i.e., for $N \geq 0$ humans, there are N according building blocks as part of a proper model for the sensor data.

In the following two examples are presented. In the first one the human is in a position which is quite standard. In the second one a very complex pose is used, which is nevertheless successfully reproduced and hence the human is recognized. Figure 12 shows a segmented infrared image and three good matches generated with our approach. Note that not only the human got recognized as indicated by an according fitness, but that it also can be localized. The input is only a 2D image, but the pseudo sensor data is based on 3D models that include information about the location of the human in 3D space.

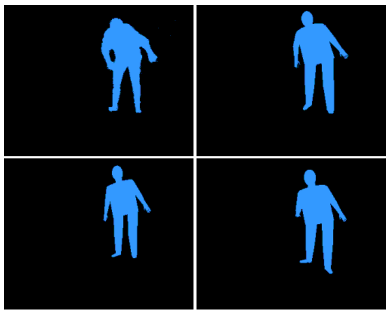


Figure 12: Complete humans can be reliably detected as shown here with an input image on the top left and three examples of best matches. Note that the input image is 2D whereas the rendered data of the matches is based on 3D models that include location data.

A second example contains a human in a very complex pose. Figure 13 shows the segmented image with a typical good match. It can be observed that some parts of the human are not matched completely correctly. What matters most is that the image is nevertheless represented by a code sniplet for a human 3D model. It is hence reliably recognized by its low fitness compared to other models that do not contain this code.

The runtimes in general allow an online recognition of humans with the onboard processing capabilities of a RugBot, which features a Pentium-M 1.2GHz processor, i.e., the images can be analyzed while the robot moves along as each generation takes about 300 msec and 50 to 300 generations are needed for reliable classification. For the general case, there is no exact estimate of the actual time it takes to detect a human or even several ones in a scene. First of all, the performance depends on which body parts can be seen. An

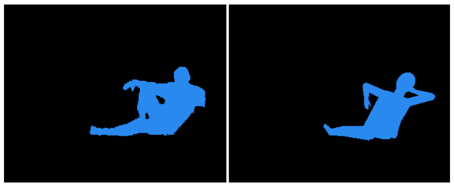


Figure 13: The segmented infrared image of a real human in a rather difficult posture and an example rendering of an evolved representation. Though there are a few minor discrepancies between the 2D image and the rendering of the 3D model, the scene is clearly recognized to contain a human. Also, its location is quite well determined.

arm can be perfectly matched in a few seconds. For the human in the complex posture, which forms a so-to-say worst case here, it took about 1.5 minutes on average to nicely match the 3D model to the 2D image. Second, the recognition is a stochastic process were increased computation time simply increases the confidence that there is indeed a human in the scene. It usually just takes a few generations, i.e., several hundred milliseconds, to transform a code sniplet representing a human such that its rendering roughly matches an image of a real human, which is indicated by low fitness values of the best individual. It strongly depends on the application whether this first rough match is considered important enough to trigger other functions or whether the evolutionary algorithm should continue first for several seconds to produce a ”perfect” fit.

3D Models of Unstructured Environments

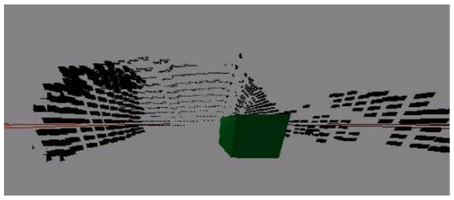


Figure 14: 3D data from an IUB rugbot generating a model of a rescue scenario with reproductive perception.

Current work in progress includes the generation of complete 3D models of unstructured environments with mobile robots using the reproductive perception approach. The underlying algorithms and principles are the same as in the application scenarios described above. The main difference is the size of the models, which requires some hierarchical organization. There is also successful work in progress where the classification and recognition methods sketched in section 3 are not only applied to humans but also to arbitrary objects.

Conclusion

Reproductive perception was introduced, an approach where models are generated, which produce data to match sensor data. This is used in several cases of spatial knowledge processing, e.g., to represent objects and their locations in the learning of eye hand coordination as well as in work on detecting and locating humans. Work in progress includes the generation of complete 3D models of unstructured environments.

References

Basri, R., and Weinshall, D. April, 1996. Distance metric between 3D models and 2D images for recognition and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18(4):465–479.

Birk, A., and Carpin, S. 2006a. Merging occupancy grid maps from multiple robots. *IEEE Proceedings, special issue on Multi-Robot Systems* 94(7):1384–1397.

Birk, A., and Carpin, S. 2006b. Rescue Robotics - a crucial milestone on the road to autonomous systems. *Advanced Robotics Journal* 20(5).

Birk, A., and J. Paul, W. 2000. Schemas and Genetic Programming. In Ritter; Cruse; and Dean., eds., *Prerational Intelligence*, volume 2. Kluwer.

Birk, A.; Markov, S.; Delchev, I.; and Pathak, K. 2006a. Autonomous Rescue Operations on the IUB Rugbot. In *IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR)*.

Birk, A.; Pathak, K.; Schwertfeger, S.; and Chonnaparamutt, W. 2006b. The IUB Rugbot: an intelligent, rugged mobile robot for search and rescue operations. In *IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR)*.

Birk, A. 1996. Learning Geometric Concepts with an Evolutionary Algorithm. In *Proc. of The Fifth Annual Conference on Evolutionary Programming*. The MIT Press, Cambridge.

Birk, A. 1998. Learning of an Anticipatory World-Model and the Quest for General versus Reinforced Knowledge. In *First International Conference on Computing Anticipatory Systems*. AIP Press.

Brown, L. G. 1992. A survey of Image Registration Techniques. *ACM Computing surveys* 24(4):325–376.

Hogg, D. 1983. Model-based vision: A program to see a walking person. *Image and Vision Computing* 1(1):5–20.

Horn, B. K. P. 1986. *Robot Vision*. MIT electrical engineering and computer science series. MIT Press, Cambridge.

J. Paul, W., and Solomonoff, R. 1995. Autonomous Theory Building Systems. In *Annals of Operations Research*. Kluwer.

Johnson, A. E. 2000. Surface Landmark Selection and Matching in Natural Terrain. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'2000)*, 413–420.

Kohlhepp, P., and Fischer, D. 2001. 3D map making by sequential registration and fusion of highly reduced submaps. In *Workshop on Autonomous Mobile Systems (AMS) 2001*.

Kohlhepp, P.; Pozzo, P.; Walther, M.; and Dillmann, R. 2004. Sequential 3D-SLAM for mobile action planning. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Mohan, A.; Papageorgiou, C.; and Poggio, T. 2001. Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, Vol.23, Iss.4, Apr 2001* 23(4):349–361.

Mount, D. M.; Netanyahu, N. S.; and Moigne, J. L. 1998. Improved algorithms for robust point pattern matching and applications to image registration. In *Proceedings of the fourteenth annual symposium on Computational geometry*. ACM Press. 155–164.

Nüchter, A.; Surmann, H.; and Hertzberg, J. 2004. Automatic Classification of Objects in 3D Laser Ranger Scans. In *Intelligent Autonomous Systems 8*. 963–970.

Oren, M.; Papageorgiou, C.; Osuna, P. S. E.; and Poggio, T. 1997. Pedestrian detection using wavelet templates. *Proc. Computer Vision and Pattern Recognition* 193–199.

Papageorgiou, C., and Poggio, T. 2000. A trainable system for object detection. *Int'l J. Computer Vision* 38(1):15–33.

Rooker, M., and Birk, A. 2006. Communicative Exploration with Robot Packs. In Noda, I.; Jacoff, A.; Bredenfeld, A.; and Takahashi, Y., eds., *RoboCup 2005: Robot Soccer World Cup IX*, volume 4020 of *Lecture Notes in Artificial Intelligence (LNAI)*. Springer. 267 – 278.

Rucklidge, W. J. 1997. Efficiently Locating Objects Using the Hausdorff Distance. *International Journal of Computer Vision* 24(3):251–270.

Sinha, P. 1994. Object recognition via image invariants: A case study. *Investigative Ophthalmology and Visual Science* 35:1735–1740.

Thrun, S. 2002. Robotic Mapping: A Survey. In Lakemeyer, G., and Nebel, B., eds., *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann.

Tung, L. H., and King, I. 2000. A Two-Stage Framework for Polygon Retrieval. *Multimedia Tools Applications* 11(2):235–255.

Vernon, D. 1991. *Machine Vision*. Englewood Cliffs: Prentice Hall.

Wren, C. R.; Azarbayejani, A.; Darrell, T.; and Pentland, A. 1997. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(7):780–785.

Yang, M.-H.; Kriegman, D.; and Ahuja, N. 2002. Detecting faces in images: a survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24(1):34–58.

Yow, K., and Cipolla, R. 1997. Feature-based human face detection. *Image and Vision Computing* 15(9):713–35.

Yuille, A. 1991. Deformable templates for face recognition. *J. Cognitive Neuroscience* 3(1):59–70.

Zhao, W.; Chellappa, R.; Phillips, P. J.; and Rosenfeld, A. 2003. Face recognition: A literature survey. *ACM Comput. Surv.* 35(4):399–458.