Locating and Tracking Objects by Efficient Comparison of Real and Predicted Synthetic Video Imagery

Damian M. Lyons*[a], D. Paul Benjamin[b]
[a]Fordham University, Robotics and Computer Laboratory, Bronx NY 10458;
[b]Pace University, Department of Computer Science, New York NY 10023

ABSTRACT

A mobile robot moving in an environment in which there are other moving objects and active agents, some of which may represent threats and some of which may represent collaborators, needs to be able to reason about the potential future behaviors of those objects and agents. In this paper we present an approach to tracking targets with complex behavior, leveraging a 3D simulation engine to generate predicted imagery and comparing that against real imagery. We introduce an approach to compare real and simulated imagery and present results using this approach to locate and track objects with complex behaviors.

In this approach, the salient points in real and imaged images are identified and an affine image transformation that maps the real scene to the synthetic scene is generated. An image difference operation is developed that ensures that the matched points in both images produce a zero difference. In this way, synchronization differences are reduced and content differences enhanced. A number of image pairs are processed and presented to illustrate the approach.

Keywords: Cognitive robotics, robot simulation, synthetic video, motion detection, computer vision.

1. INTRODUCTION

Consider the task of having a robot intercept a ball rolling on the ground (the robot soccer problem). A behavior-based approach including visual tracking of the ball can yield a robust solution (e.g., [8][10]). However, if we complicate the problem by considering the ball to be moving towards a wall or another agent then the situation becomes much more difficult. The dynamics used in tracking a ball typically does not include information about bouncing off walls or other agents. Although a fast tracking system may reacquire the ball target after the bounce, it certainly will not be able to predict the bounce. Hence any action that the robot takes before the bounce will be predicated on the ball continuing its observed path. This puts the robot in the position of always playing 'catch-up' with the ball instead of accurately predicting where the ball will be and moving there. Although we have portrayed this issue in terms of the robot soccer problem, this same issue arises whenever a robot is operating in a complex dynamic environment, for example, an urban search and rescue robot moving on a semi-stable pile of rubble.

We introduce an approach to modeling the behavior of a complex, dynamic environment that attempts to maintain the simplicity of a behavior-based robotics approach. We will exploit a 3D modeling system called OGRE [7], an open-source rendering engine for building 3D games, for modeling the appearance of objects around the robot. OGRE can be integrated with ODE (the Open Dynamics Engine) to include the modeling of the physics of objects. Returning to the robot soccer problem, if a simulated ball is moving towards a simulated wall in an OGRE/ODE simulation, then the simulation can certainly predict the position of the ball after the bounce. Informally, the simulation can act as the 'imagination' of the robot, allowing it to carry out a particular kind of thought experiment: allowing the simulated world to run faster than the real world for the purposes of prediction. In Section 3 we describe our approach to integrating simulation and observation by comparing real visual input with the graphical, synthetic video generated by a simulation to determine how well observations match simulated expectations. We present experimental results in Section 4 for several different scenes and for different conditions, illustrating how this approach can be used to locate unexpected objects and the lack of expected objects.

2. LITERATURE REVIEW

The growth of the video game industry has contributed to the ready availability of good 3D simulation environments,

and a number of projects have involved integrating robot simulation and control environments, e.g., [1][5]. That approach is focused on providing ready interleaved access to simulation or robot control to a programmer. In contrast, our use of simulation is as an intrinsic part of the robot control itself, not as an aid to the programmer. In the assembly and task planning field, the integration of simulation into the reasoning process has been investigated [12]; however, the integration was achieved by modeling the state of the environment. Our objective is to integrate simulation while maintaining the generality and robustness of a behavior-based approach, not sharing the simulation state information.

In previous work [2] we have described an approach to this problem based on comparing the visual input of the robot with imagery generated from a dynamic 3D world model and directing discrepancies to the robot's SOAR-based cognitive system. Integration of Video and 3D imagery has been considered in applications such as predictive teleoperation [3]. Our problem is different in that it requires comparing the synthetic and real imagery to look for differences between actual and predicted object behavior. Comparing the synthetic imagery from the world model with imagery from the vision system poses a number of problems [4], including synchronization differences: differences in the camera poses, in the scene lighting, and in the colors and textures; as well as content differences: differences in the number and type of object shown and differences in the predicted object behavior. All these issues mean that a simple difference operation between real and synthetic images is not very effective.

3. COMPARING REAL AND SYNTHETIC SCENES

In this paper, we focus on just the task of *comparing real and synthetic video*. Figure 1, however, shows the block diagram of our overall system [3]: The real and synthetic images of the scene as viewed by the robot are compared. If the scenes are considered the same but from different viewpoints, then the viewpoint of the camera in the simulation is changed, and the simulation generates an image taken by the camera at the new location. If an unexpected object is seen in the real image, an object is introduced at the corresponding position in the simulated scene. The region of the real image responsible for the difference is used as video texture on the object and a new synthetic image generated. The information on whether there is no difference, an unexpected object, or an object missing between the image pairs is made available to action planning [2]. This loop of difference detection and simulation modification is used to keep the simulation synchronized to the observed environment. For prediction purposes, the simulation can be allowed to 'fast forward' in time, so that the expected position, for example, of a target can be calculated and then compared to observations.

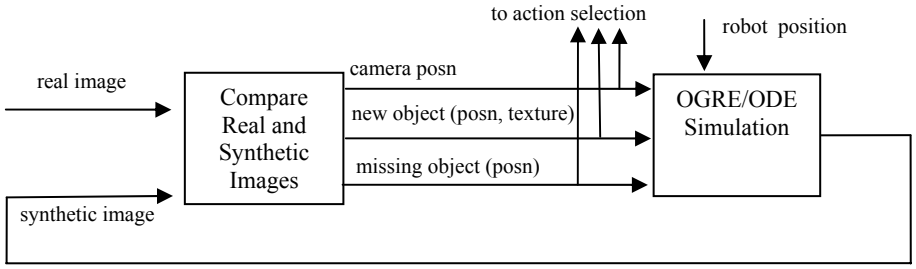


Figure 1: Block diagram of the loop integrating simulation and observation

Fig. 2 shows a real (2(A)) and synthetic (2(B)) view of the same scene taken with the artificial camera at approximately the same location and orientation as the camera in the real scene.

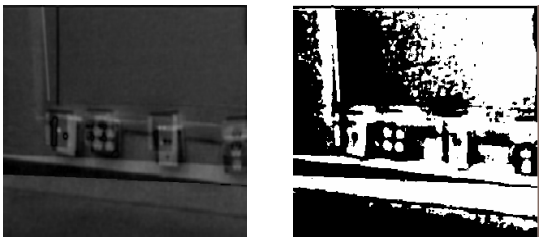
(A) (B)

Figure 2: Real (A) and synthetic (B) views of the same scene from approximately the same position and orientation

The view presented here is a close up of one wall of a room. The scene is also modeled graphically using OGRE. Sections of the graphical scene have been tiled with video texture manually extracted from Fig. 2(A). The use of video texture should make it easier to directly compare the real image and synthetic image to answer the following questions:

1. Do they represent the same scene from the same viewpoint?
2. Do they represent the same scene from slightly different viewpoints?
3. Do they represent the same scene but with some number of different objects?
4. Do they represent different scenes?

Let I_s be the simulated image and I_r be the real image. Fig. 3(A) shows the absolute image difference $|I_s - I_r|$ and Fig. 3(B) is a thresholded absolute difference. The images show substantial differences because the real and simulated camera positions are not identical and errors are introduced by the texture extraction and tiling. However, we would like to be able to determine that these are views of the same scene, albeit from slightly different camera positions.



(A) (B)

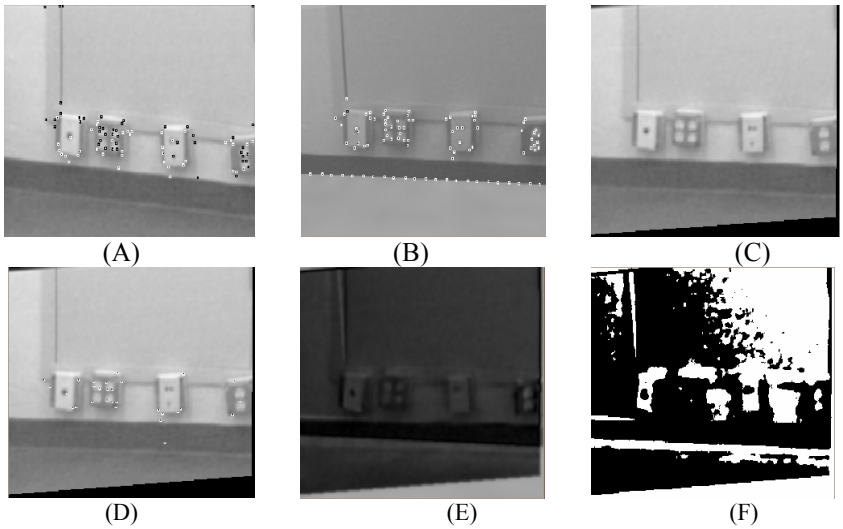
Figure 3: Absolute difference (A) and thresholded difference between real and synthetic images from Fig. 2.

3.1 Alignment of Synthetic and Real Images

If we just consider the camera misalignment issue, we can approximate the registration between the real and synthetic images by an affine transformation. If $p_s=(x_x, y_s)$ is a point on I_s and $p_r=(x_r, y_r)$ is a point on I_r then we can say:

$$p_r = A\, p_s + b \qquad (1)$$

where A is a 2×2 rotation matrix and b a translation. If points of correspondence can be established between the real and synthetic images, then the affine parameters A and b can be estimated. Using the efficient corner detection library of Trajkovic & Hedley [10] corners were labeled in the real and synthetic images (e.g., Fig. 4(A, B) for the scene of Fig. 2). The RANSAC algorithm [5] was used to estimate the affine parameters A and b. Fig 4(C) shows the real image transformed for registration with the synthetic image, I'_r and Fig. 4(D) shows the points used to estimate the transform.



(A) (B) (C)

(D) (E) (F)

Figure 4: Corners detected in real (A) and synthetic (B) scenes, real scene affine warped to synthetic scene (C) using matched points (D), absolute difference (E) and thresholded difference (F) with warped real and synthetic images.

The difference operation $| I_s - I'_r |$ is shown in Fig. 4(E) and the thresholded result in Fig. 4(F). Comparing Fig. 3(B) and Fig. 4(E) it can be seen that some of the sources of the difference error have been resolved but not eliminated. There is less difference error on the lines and edges on the wall and floor, but the affine registration on its own is not sufficient.

3.2 Calculating the Match-Mediated Difference Mask

The affine transformation brings corresponding objects in the real and synthetic images approximately into registration. However, there are still differences caused by the quality of the texture mapping or simulated surface color. To address this, we will make the assumption that the image area around a point used to estimate the affine registration should be similar in both images. The better two matched points correspond between real and synthetic images, the more we will assume the two images should be similar.

Let p_r and p_s be two matched points in the real and synthetic images. We will consider one of the images to be the primary image and construct the difference image in those image coordinates. In this paper we consider the synthetic image to be the primary image. Each match point p' in the set of match points P will be in the image coordinates of the synthetic image. Its corresponding point in the real image, $m(p')$, will be given by the affine transform in eq. (1):

$$m(p') = A\,p' + b$$

We will place a normalized Gaussian at each point p' in the set of match points P and sum these over the image to create an image mask whose values correspond to the proximity of the image pixel to adjacent match points:

$$\frac{1}{|P|}\sum_{p'\in P}\frac{1}{S_{p'}}e^{\frac{(p-p')^2}{2v}} \qquad (2)$$

where $S_{p'}$ is the sum of the Gaussian for p' over the entire image I and v a small, fixed variance parameter:

$$S_{p'} = \sum_{p\in I} e^{\frac{(p-p')^2}{2v}} \qquad (3)$$

However, this doesn't account for the fact that some matches are of better quality than others. If p is a point in the set of match points P, we define the match error $e(p)$ to be the distance between the two matched points p and $m(p)$ in the primary image coordinates:

$$e(p) = |p - m(p)|$$

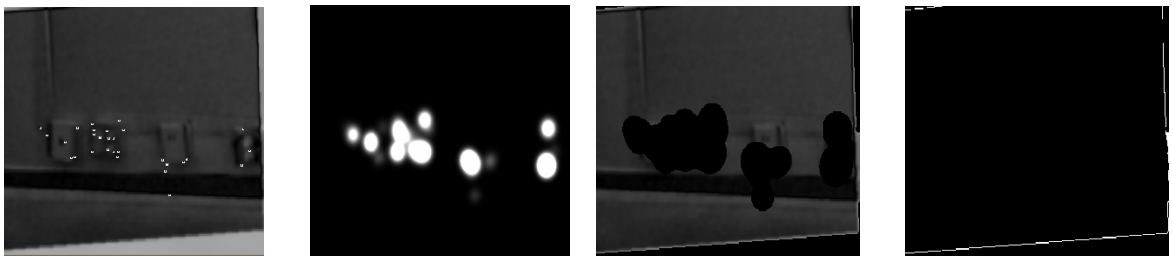
We define the normalized match quality $q(p)$ to be the inverse of the match error normalized by the sum of all match errors:

$$q(p) = \frac{1}{e(p)\sum_{p'\in P}\frac{1}{e(p')}} \qquad (4)$$

Eq. (4) is a measure of the quality of match p with respect to all the other matches, and we use this as a coefficient for the Gaussians in Eq. (2) to generate the *match-mediated difference mask* I_m :

$$I_m(p) = \frac{1}{|P|}\sum_{p'\in P}\frac{q(p')}{S_{p'}}e^{\frac{(p-p')^2}{2v}} \qquad (5)$$

Figure 5(A) shows the match points for the running example superimposed on the absolute difference image $|I_s - I'_r|$. Fig. 5(B) shows the resultant gray level match-mediated difference mask.



(A) (B) (C) (D)

Figure 5: Absolute difference of warped real and synthetic images with overlaid match points (A) and match mediated difference mask (B) calculated from (A). Match mediated absolute difference (C) and thresholded difference (D).

To calculate the match-mediated difference image using the match-mediated difference mask eq. (5) we divide each point in the difference image by the corresponding point in the mask:

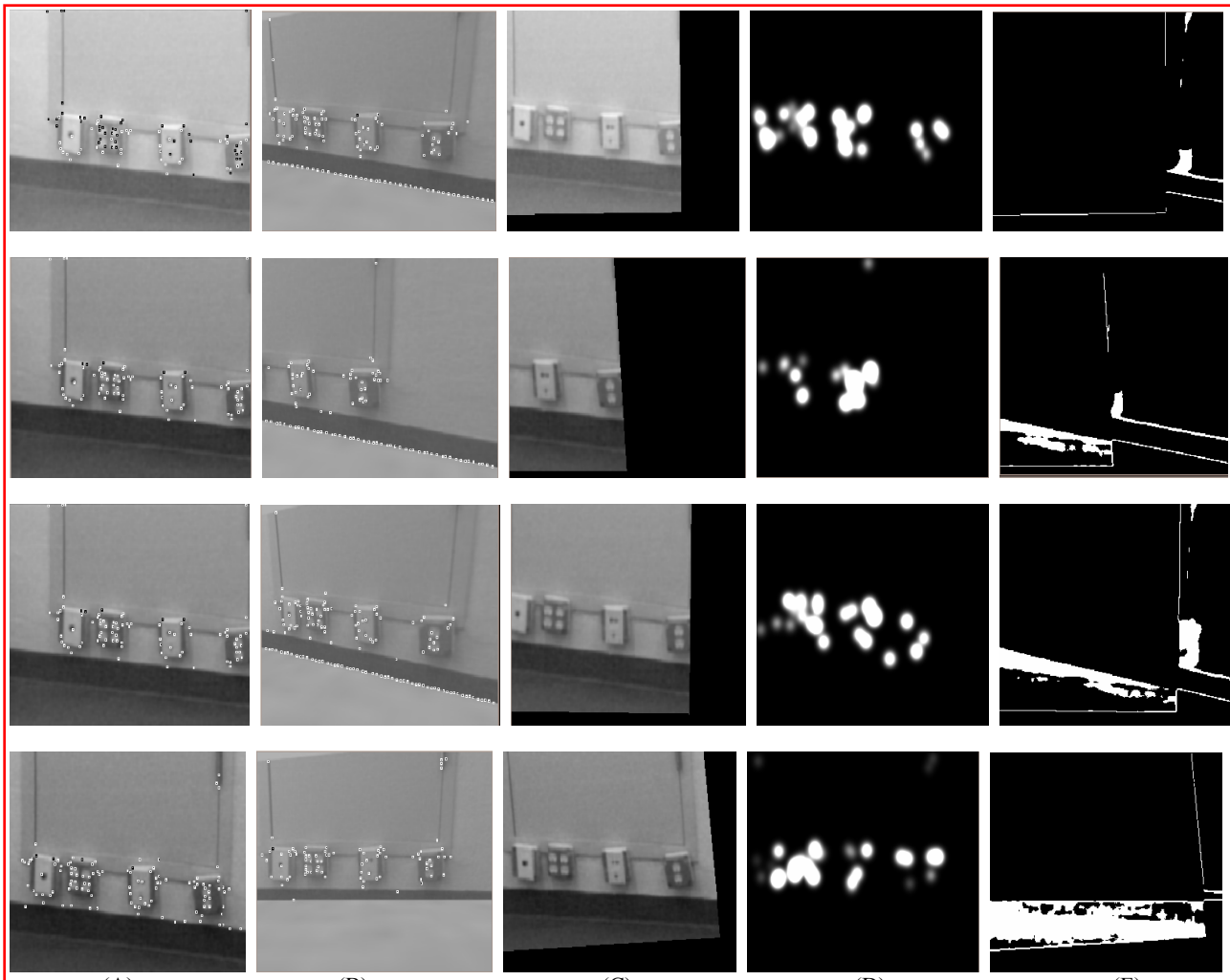
$$I_d(p) = \frac{|I_s(p) - I'_r(p)|}{I_m(p)} \qquad (6)$$

and the result for the running example is shown in Fig. 5(C) and thresholded in Fig. 5(D). (The thresholds used throughout are the same for all images). The resulting difference image shows only the edge of the common region in the primary (synthetic) and affine transformed secondary (real) image, allowing us to finally say that both images are of the same scene from different viewpoints as given by A and b from eq. (1).

4. EXPERIMENTS

In this section we show the results of experiments using the match-mediated difference approach to detect whether real and synthetic images show the same scene, a scene with a new object, or a scene with a missing object. Fig. 6 shows four pairs of real and synthetic images. The first three pairs attempt to match the same real scene with slightly different views of the synthetic scene. The first and fourth pair attempt to match the same synthetic scene with slightly different views of the real scene. In each case, the image pair is shown in columns (A) and (B) overlaid with the corner point results; column (C) shows the affine transformed real image; column (D) shows the gray-level match-mediated mask; and, column (D) shows the thresholded match-mediated absolute difference image. In the final column, the only part of the difference image that is valid is the overlap between the synthetic and affine transformed real image and the boundary is typically visible.

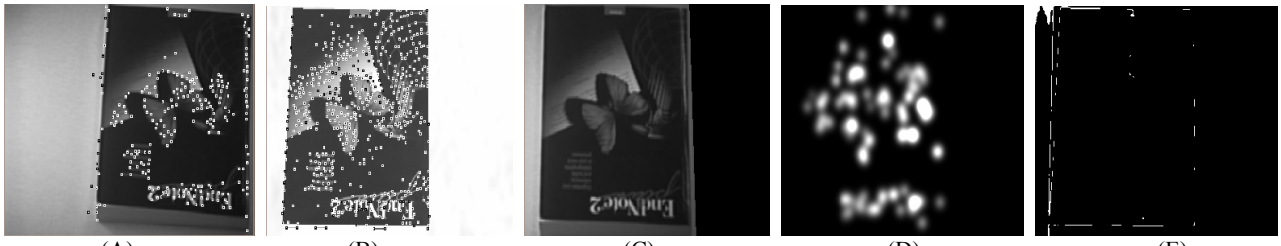
The first image pair shows the ideal result, the difference image is empty in the overlap region. However, in the remaining three rows, the image is blank for most of the overlap region, except for the floor. The region of the image with the most complicated geometric features remains blank because of the math-mediated difference mask. The floor in the synthetic scene is visually quite different from the floor in the real scene, the result of imperfect manual texture collection from the real scene and mapping in the simulated scene. However, the image scope of the difference is sufficiently small (unlike the large areas of difference in Fig 3(B)) that we expect that we will be able to use this to iteratively refine the texture in the simulated scene and reduce the observed difference via the loop shown in Fig. 1.



(A) (B) (C) (D) (E)

Figure 6: Examples of the scene in Fig. 2 but with the simulated camera moved: Cols. A and B are the real and synthetic images with corner points; col. C is the affine transformation image; col. D is the match-mediated mask; and, E the match-mediated difference.

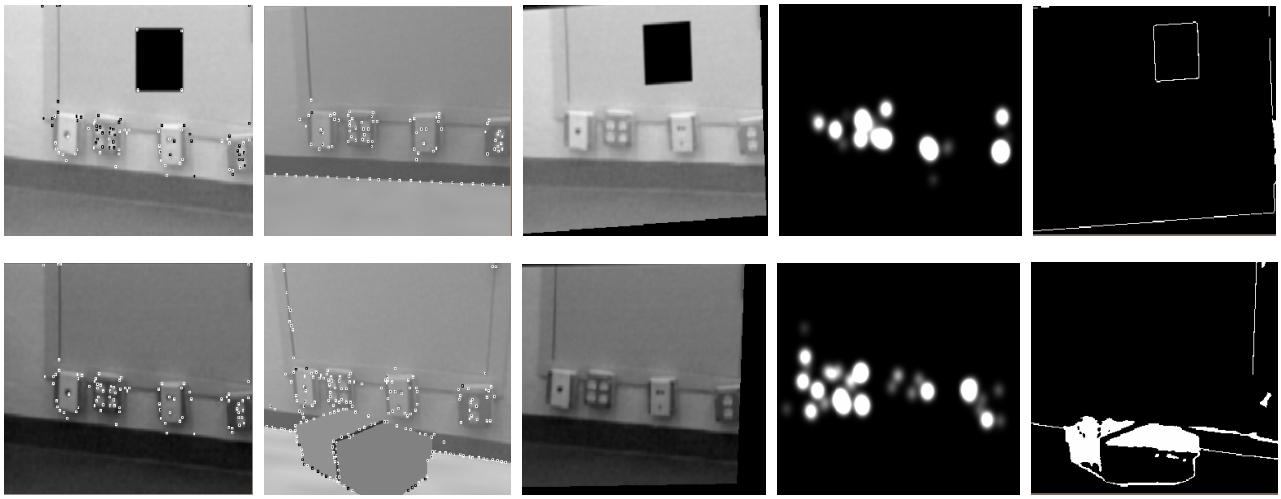
All the examples so far are of image pairs that should produce no difference. To be useful, this approach should preserve differences that are due to new objects in either real or synthetic image. Our convention is to consider an object in the real image but not in synthetic image as an *unexpected object*, and an object in the synthetic image but not the real image as a *missing, expected object*. In Fig. 8, the top line shows the same experiment presented as the running example in Section 3, except a black square has been artificially drawn on the back wall of the real image. The process of



(A) (B) (C) (D) (E)

Figure 7: Book example images: Cols. A and B are the real and synthetic images with corner points; col. C is the affine transformation image; col. D is the match-mediated mask; and, E the match-mediated difference.

estimating the affine transform is the same as for the original experiment. However, since there are no matches on the black square – as it appears in only one image of the image pair – the match-mediated difference mask contains zero or small values in the vicinity of this feature (Fig. 8(D)). Hence the feature is preserved when the difference operation (eq. (6)) is evaluated, showing up clearly in Fig. 8(E).



(A) (B) (C) (D) (E)

Figure 8: Example of a scene with an expected object missing: object is in synthetic image and not in real image: Cols. A and B are the real and synthetic images with corner points; col. C is the affine transformation image; col. D is the match-mediated mask; and, E the match-mediated difference.

The second row of Fig. 8 shows a box introduced into the synthetic scene. The scene was generated by making a graphical model of a box roughly similar in appearance to the box in Fig. 9(A) and placing on the floor close to the wall in the 3D Ogre scene model. Because of the proximity of the box to corner features used as match points, the match-mediated difference mask does partially overlap the region of the image where the box is. Nonetheless, the thresholded result extracts the majority of the box as a valid difference region. In this case, we would consider this a missing expected object.

(A) (B) (C) (D) (E)

Figure 9: Example of a scene with an unexpected object - object is not in synthetic image but is in real image: Cols. A and B are the real and synthetic images with corner points; col. C is the affine transformation image; col. D is the match-mediated mask; and, E the match-mediated difference.

Figure 9 shows an example of an unexpected object. The corner points on the box contribute minimally to the affine transform and to the match-mediated difference mask. The thresholded result does indeed show the box against the floor; however, so much of the floor also shows up that it is difficult to identify the box as an expected but missing object. Our approach here, as in the last few examples in Fig. 6, is to use the difference region extracted as a mask to extract floor texture from the video via the loop shown in Fig. 1. With better floor texture, we expect the box to be separable.

5. SUMMARY

In this paper, we introduced an approach to integrating a 3D simulation system based on OGRE/ODE with the visual processing module of a robot control system. The objective is to use the simulation system to model complex phenomena in the environment. The simulation is integrated into the robot control architecture in a novel way so that the many of the advantages of a behavior based control approach can be maintained; the simulation presents its output as an alternate visual input – the 'expected' visual scene. We present a novel technique, the match-mediated motion difference, for comparing real and synthetic images that takes into account that the two images may be taken from different camera viewpoints, may contain some differences in color and texture, and may contain different objects.

The approach works for as long a sufficient number of corner points can be extracted from each image and an affine transform can be found to match the images. In the case that an affine transformation cannot be found, the images are considered too different to compare. Another constraint is that any real regions of difference are sufficiently distinct from the points used to make the affine transform. This constraint may result in the edges of objects being clipped, as for example in Fig. 8(E) second row.

All the examples here started with a manual extraction of texture for the simulation. A major avenue of future work will investigate the automation of the loop in Fig. 1 for updating the simulation by extracting texture from regions identified as difference regions. For example, the floor in Fig. 9(E) would be identified as a difference, the difference region used as a mask to extract texture from the real video, and the texture added in to the simulation. This loop should converge by incremental identification of differences, extraction of texture, and updated of simulation model to a zero difference image.

ACKNOWLEDGEMENTS: The OGRE simulation for the lab well examples was done by Sirhan Chaudry, and for the book examples by Mohamed Shaheen Ali.

REFERENCES

[1] Albrecht, S., J. Hertzberg, K. Lingemann, A. Nüchter, J. Sprickerhof, and S. Stiene. "Device Level Simulation of Kurt3D Rescue Robots," *Proceedings of the 3rd International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster (SRMED 2006)*, June (2006)

[2] Benjamin, D.P., Lonsdale, D., and Lyons, D.M., "Embodying a Cognitive Model in a Mobile Robot," Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision, Boston, October (2006).

[3] Benjamin, D.P., Achtemichuk, T.., and Lyons, D.M., "Obstacle Avoidance using Predictive Vision based on a Dynamic 3D World Model," Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision, Boston, October (2006).

[4] Burkert, T., and Passig, G., "Scene Model Acquisition for a Photo-Realistic Predictive Display," Proceedings of the Third International Conference on Humanoid Robots, October (2004).

[5] Diankov, R., and J. Kuffner, "*OpenRAVE: A Planning Architecture for Autonomous Robotics*," Tech. report CMU-RI-TR-08-34, Robotics Institute, Carnegie Mellon University, July (2008).

[6] Fischler, M. and Bolles, R. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. of the ACM,* **24**: 381–395, June (1981)

[7] Junker, G., *Pro OGRE 3D Programming*, Apress, (2006).

[8] Mantz, F., Pieter Jonker, "Behavior Based Perception for Soccer Robots," in: Goro Obinata, Ashish Dutta, Nagoya University (Eds),*Vision Systems* Advanced Robotic Systems, Vienna, Austria, April (2007).

[9] Rushmeier, H., Greg Ward, Christine Piatko, Phil Sanders and Bert Rust, "Comparing Real and Synthetic Images: Some Ideas about Metrics," Eurographics Workshop on Rendering, Dublin, Ireland, (1995).

[10] Shen, W., Jafar Adibi, Rogelio Adobatti, Bonghan Cho, Ali Erdem, Hadi Moradi, Behnam Salemi, Sheila Tejada. "Autonomous Soccer Robots". Lecture Notes in Computer Science 1395, Springer-Verlag, (1998).

[11] Trajkovic, M., Hedley, M., "Fast Corner Detection," *Image and Vision Computing*, (16) pp. 75-87 (1998)

[12] Xiao, J., Zhang, L., "A Geometric Simulator SimRep for Testing the Replanning Approach toward Assembly Motions in the Presence of Uncertainties," IEEE Int. Symp. Assembly and Task Planning, (1995).