Augmented Reality for Robot Development and Experimentation

Mike Stilman,[1] Philipp Michel,[1] Joel Chestnutt,[1]
Koichi Nishiwaki,[2] Satoshi Kagami,[2] James J. Kuffner[1,2]

[1] The Robotics Institute
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213, USA
{mstilman, pmichel, chestnutt, kuffner}@cs.cmu.edu

[2] Digital Human Research Center, National Institute of
Advanced Industrial Science and Technology (AIST)
2-41-6 Aomi, Koto-ku, Tokyo, Japan 135-0064
{k.nishiwaki, s.kagami}@dh.aist.go.jp

Abstract

The successful development of autonomous robotic systems requires careful fusion of complex subsystems for perception, planning, and control. Often these subsystems are designed in a modular fashion and tested individually. However, when ultimately combined with other components to form a complete system, unexpected interactions between subsystems can occur that make it difficult to isolate the source of problems. This paper presents a novel paradigm for robot experimentation that enables unified testing of individual subsystems while acting as part of a complete whole made up of both virtual and real components. We exploit the recent advances in speed and accuracy of optical motion capture to localize the robot, track environment objects, and extract extrinsic parameters for moving cameras in real-time. We construct a world model representation that serves as ground truth for both visual and tactile sensors in the environment. From this data, we build spatial and temporal correspondences between virtual elements, such as motion plans, and real artifacts in the scene. The system enables safe, decoupled testing of component algorithms for vision, motion planning and control that would normally have to be tested simultaneously on actual hardware. We show results of successful online applications in the development of an autonomous humanoid robot.

1 Introduction

As robotics researchers strive to develop more sophisticated autonomous systems, thorough testing of various interconnected hardware and software components for perception, planning, and control becomes increasingly difficult. A common paradigm in experimental robotics is the two stage process of virtual simulation and real world experimentation. The purpose of virtual simulation is to find critical system flaws or software errors that would cause failures or other undesirable behaviors during an actual robot experiment. In the context of complex systems such as mobile manipulators and humanoid robots, virtual environments allow the researcher to independently evaluate critical subsystems. Many software tools are available for dynamic simulation and visualization. However, when robots are put to the test in real environments these tools are only used offline for processing the data of an experiment. We propose an alternate paradigm for real-world experimentation that utilizes a real-time optical tracking system to form a complete hybrid real/virtual testing environment.

Our proposed system has two objectives: to present the researcher with a ground truth model of the world and to introduce virtual objects into actual real world experiments. To see the relevance of these tools, consider an example of how the proposed system is used in our laboratory. A humanoid robot with algorithms for vision, path planning and ZMP stabilization is given the task of navigation in a field of obstacles. During an online experiment, the robot unexpectedly contacts one of the obstacles. Did our vision system properly construct a model of the environment? Did the navigation planner find an erroneous path? Was our controller properly following the desired trajectory? A ground truth model helps resolve ambiguities regarding the source of experimental failures by precisely identifying the locations of the obstacles and the robot. Just as in

simulation, we can immediately determine whether the vision algorithm identified the model, or whether the controller followed the trajectory designed by the planner. In some cases, we can avoid the undesired interaction entirely. Having established a correspondence between virtual components such as environment models, plans, intended robot actions and the real world, we can then visualize and identify system errors prior to their occurrence.

In this paper, we describe the implementation of the hybrid experimental environment. We develop tools for constructing a correspondence between real and virtual worlds. Using these tools we find substantial opportunities for experimentation by introducing virtual obstacles, virtual sensors and virtual robots into a real world environment. We describe how adding such objects to an experimental setting aids in the development and thorough testing of vision, planning and control.

2 Related Work

Preliminary testing of a robot system in a simulation environment offers many advantages to direct implementation. Not only is virtual simulation safe, but also it enables the researcher to observe the complete state of the virtual world, interact with it and visualize the performance of each system component under a variety of conditions. In the domain of complex robot platforms such as humanoid robots, the OpenHRP [1] simulation engine has become a common tool for modeling dynamics and testing the performance of controllers. Other simulations [2] focus on the kinematics and geometry for testing higher level planning and vision components. Research by Khatib et. al. [3] adds the ability to haptically interact with the virtual environment. Yet, despite the development of fast and precise algorithms for dynamic modeling [4, 5, 6, 7, 8] purely virtual simulations are limited to approximating the real world. Common assumptions such as rigid body dynamics and perfect vision make virtual experiments preliminary to hardware experimentation. Final modifications are most often performed in the real world without any of the advantages of a virtual system. In the best case, these final tests involve only minor parameter tuning. These tests, however, may also reveal unexpected subsystem interactions that require nontrivial software modifications. In the worst case, severe system failures or dangerous accidents may occur.

To minimize these risks, *hardware in the loop* simulation paradigms have been utilized. This approach is most common for experiments in aeronautics and space robotics. For instance, Carufel presents a controller designed for zero-gravity applications [9]. The system can still be tested on hardware in a laboratory setting if an additional controller acts to compensate for the effects of gravity. Another instance of this can be seen in ViGWaM [10], where a vision system designed to function on a walking machine is tested separately from an actual biped, enabling concurrent development of both hardware and software. Experiments are performed with a wheeled mobile manipulator that imitates the gyration of a biped walker. Like these examples, most hardware in the loop systems are designed around specific applications. Our goal is to present a general augmentation scheme that can be used in a variety of experimental contexts.

The field that concentrates on combining real and virtual worlds is augmented reality. Azuma [11, 12], describes recent developments in augmentation with overviews of tracking, overlays and applications. Works in augmented teleoperation are particularly interesting. The results of Milgram, Drascic et al. [13] demonstrate the use of virtual overlays for robot teleoperation to design and evaluate robot plans by visually combining video of the robot with an overlaid model. Typically these systems are designed for fixed cameras and manipulators. Furthermore, they focus on human plan development, rather than human supervision of autonomous algorithms. More complex scenarios with autonomous mobile manipulators require both the robot and the researcher's view to freely move anywhere in the environment.

One alternative to a fixed view is visual registration of features in a camera view as examined by Kutulakos [14] and Uenohara and Kanade [15]. While these methods are successful in selected tasks, Dorfmuller points out that speed, robustness and accuracy of a tracking system can be enhanced by binocular cameras and hybrid tracking by the use of markers [16]. In particular, he advises the use of retroreflective markers. Some systems use LED markers [17], while others combine vision-based approaches with magnetic trackers [18].

We observe that large area coverage of accurate object localization and tracking is typically performed with a motion capture system that consists of an array of cameras. Recently, motion capture systems such as those manufactured by Motion Analysis Corp. allow the co-registration and localization of markers across a dozen cameras at a rate of 480Hz.

This array of cameras provides the speed and accuracy described by Dorfmuller [16], as well as complete coverage of the experimental environment. In our work, *view cameras* are not used for scene registration. They are parts of the scene that are tracked along with other objects. In the remainder of this paper, we will explore the versatility of our augmented system and its applications to robot experimentation.

3 Overview

3.1 Experimental Setting

To construct a hybrid real/virtual environment, we instrumented our lab space with the Eagle-4 Motion Analysis motion capture system [19]. The environment also contains cameras and furniture objects. Our experiments focused on high level autonomous tasks for the humanoid robot HRP-2. For instance, the robot navigated the environment while choosing foot locations to avoid obstacles [20] and manipulated obstacles to free its path [21]. We partitioned these experiments according to the subsystems of vision, planning and control to provide a general groundwork for how a hybrid real/virtual testing environment can be used in a larger context of research objectives.

3.2 Technical Details

The Eagle-4 system consists of eight cameras, covering a space of 5×5 meters to a height of 2 meters. Distances between markers that appear in this space can be calculated to 0.3% accuracy. In our experiments, the motion capture estimate of the distance between two markers at an actual distance of 300mm has less than 1mm error.

In terms of processing speed, we employ a dual Xeon $3.6GHz$ processor computer to collect the motion capture information. The EVa Real-Time Software (EVaRT) registers and locates 3D markers at maximum rate of 480Hz with an image resolution of 1280×1024. When larger numbers of markers are present, the maximum update speed decreases. Still, when tracking approximately 60 markers the lowest acquisition rate we used was 60Hz. Marker localization was always performed in real-time.

4 Ground Truth Modeling

4.1 Reconstructing Position and Orientation

EVaRT groups the markers attached to an object. The markers can be expressed as a set of points $\{\mathbf{a}_1, ..., \mathbf{a}_n\}$ in the object's coordinate frame $\mathcal{F}$. We refer to this set of points as the object *template*.

Under the assumption that a group of markers is attached to a rigid object, any displacement of the object corresponds to a rigid transformation $\mathcal{T}$ of the markers. A displaced marker location $\mathbf{b}_i$ can be expressed with a homogeneous transform:

$$\mathbf{b}_i = \begin{bmatrix} & \mathcal{R} & & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{a}_i \tag{1}$$

During online execution, EVaRT uses distance comparisons to identify groupings of markers, as well as the identities of markers in these groupings. We are then interested in the inverse problem of finding a transform $\mathcal{T}$ that aligns the template marker locations with those found in the scene by motion capture.

In order to find the transformation of an object in our scene, we follow a two step procedure:

1. Using the set of markers currently visible, find the centroids ($\mathbf{c}_a$ and $\mathbf{c}_b$) of the template marker points and the observed marker locations. Estimate the translational offset:

   $$\hat{\mathbf{t}} = \mathbf{c}_b - \mathbf{c}_a$$

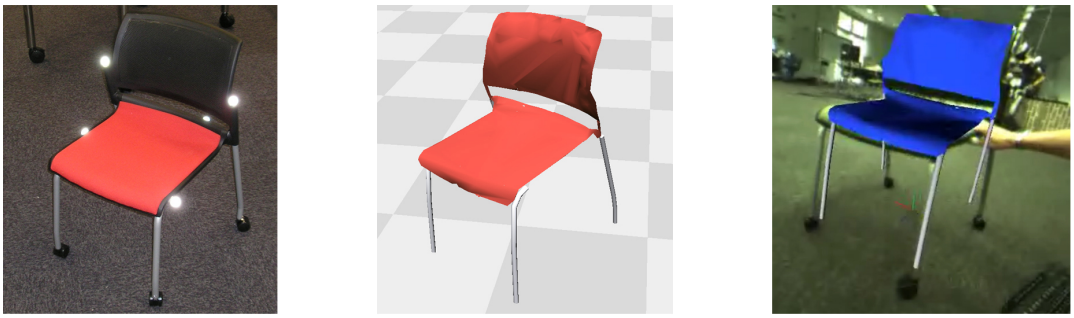
   We define $\mathbf{b}'_i = \mathbf{b}_i - \hat{\mathbf{t}}$. This places the coordinate frames of perceived and template markers at a common origin.

2. Next, we define a linear system that represents the orientation of our object. For a 3×3 matrix of the form $\mathcal{R} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{bmatrix}^T$ we can express the system as follows:

   $$\begin{bmatrix} \mathbf{a}_1^T & & 0 \\ & \mathbf{a}_1^T & \\ 0 & & \mathbf{a}_1^T \\ & \cdots & \\ \mathbf{a}_n^T & & 0 \\ & \mathbf{a}_n^T & \\ 0 & & \mathbf{a}_n^T \end{bmatrix} \begin{bmatrix} \hat{\mathbf{r}}_1 \\ \hat{\mathbf{r}}_2 \\ \hat{\mathbf{r}}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{b}'_1 \\ \mathbf{b}'_2 \\ \vdots \\ \mathbf{b}'_n \end{bmatrix} \tag{2}$$

   We solve this system for $\hat{\mathcal{R}}$ online using LQ decomposition.

Combining the translation $\hat{\mathbf{t}}$ and the $\hat{\mathcal{R}}$ matrix yields a 12 DOF affine transformation for the object. At this time, we do not enforce rigidity constraints. Even for a system with only four markers, the accuracy of motion capture described in Section 3 yields negligible shear and scaling in the estimated transformation.



(a) (b) (c)

Figure 1: (a) Real chair with retroreflective markers illuminated. (b) 3D model of chair as recoverd by a laser scanner. (c) Virtual chair is overlayed in real-time. Both the chair and the camera are in motion.

Since our matrix inversion does not rely on the values of observed coordinates, we could potentially pre-compute this aspect of the algorithm. However, observe that during online execution, some markers may be occluded from motion capture. In this case, the algorithm must be performed only on the visible markers. When markers are occluded, their corresponding rows in Equation 2 must be removed. Furthermore, the centroids used in computing the translation $\hat{\mathbf{t}}$ must be the centroids of the visible markers and their associated template markers.

4.2 Reconstructing Geometry

The transformation of a rigid body's coordinate frame tells us the displacement of all points associated with the body. To reconstruct the geometry of a scene, we need to establish the geometry of each object in its local reference frame.

In our work, we have chosen to use 3D triangular surface meshes to represent environment objects. We constructed preliminary meshes using a Minolta VIVID 910 non-contact 3D laser digitizer. The meshes were manually edited for holes and automatically simplified to reduce the number of vertices.

Figure 1 demonstrates the correspondence between a chair in the lab environment and its 3D mesh in our visualization. Applying the algorithm in the previous section, we are able to continuously re-compute the transformation of a lab object at a rate of 30Hz. The virtual environment can then be updated in real-time to provide a visualization of the actual object's motion in the lab.

4.3 Real and Virtual Cameras

Section 4.1 described a method for identifying the position and orientation of a rigid body in a scene using motion capture. The rigid objects could be obstacles as in Section 4.2, the bodies composing the robot, or sensors such as cameras. In this section we consider the latter case of placing a camera in the viewable range of motion capture. We show that tracking a camera lets us to establish a correspondence between objects in the ground truth model and objects in the camera frustum.

As with other rigid bodies, the camera is outfitted with retro-reflective markers that are grouped in EVaRT and then tracked using our algorithm. The position and orientation of the camera computed from motion capture form the extrinsic camera parameters. The translation vector $\mathbf{t}$ corresponds to the world coordinates of the camera's optical center and the 3×3 rotation matrix $\mathcal{R}$ represents the direction of the optical axis. Offline camera calibration using Tsai's camera model [22] is performed once to recover the the 3×3 upper triangular intrinsic parameter matrix $\mathcal{K}$. Incoming camera images can then be rectified on the fly. The extrinsic and intrinsic parameters allow us to recover the full camera projection matrix $\mathcal{M}$:

$$\mathcal{M} = \mathcal{K} \begin{bmatrix} \mathcal{R} & \mathbf{t} \end{bmatrix} \tag{3}$$

$\mathcal{M}$ uniquely maps a scene point $\mathbf{P} = (x, y, z, 1)^T$ to a point on the image plane $\mathbf{p} = (u, v, 1)^T$ via the standard homogeneous projection equation:

$$\mathbf{p} \equiv \mathcal{M}\mathbf{P} \tag{4}$$

From Section 4.2, we can recover not only the locations of motion capture markers but also any points

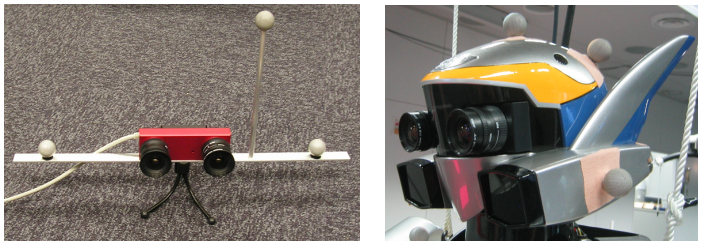


Figure 2: Marker equipped firewire camera bodies used for localization: metal frame *(left)* and humanoid head *(right)*. Only one camera from the stereo pair is used.

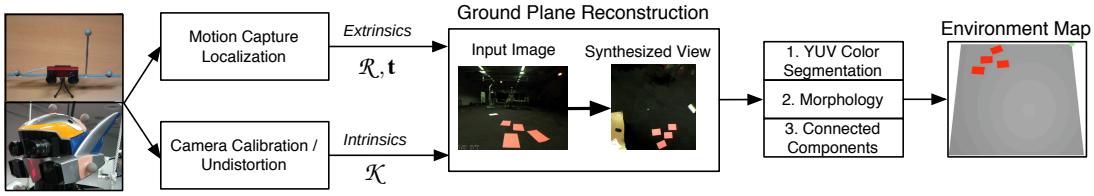


Figure 3: Overview of the image-based reconstruction process. An environment map of obstacles on the floor is constructed from a calibrated, moving camera localized using motion capture.

that compose the surface mesh of a tracked object. Transforming these points using Equation 4, we can identify projections for all triangles in the surface mesh onto the image plane. Equivalently, we can use existing 3D display technology such as OpenGL to efficiently compute surface models as they would appear in the camera projection. Overlaying the virtual display on the camera display creates the a correspondence between the camera view and the ground-truth motion capture view as shown in Figure 4.

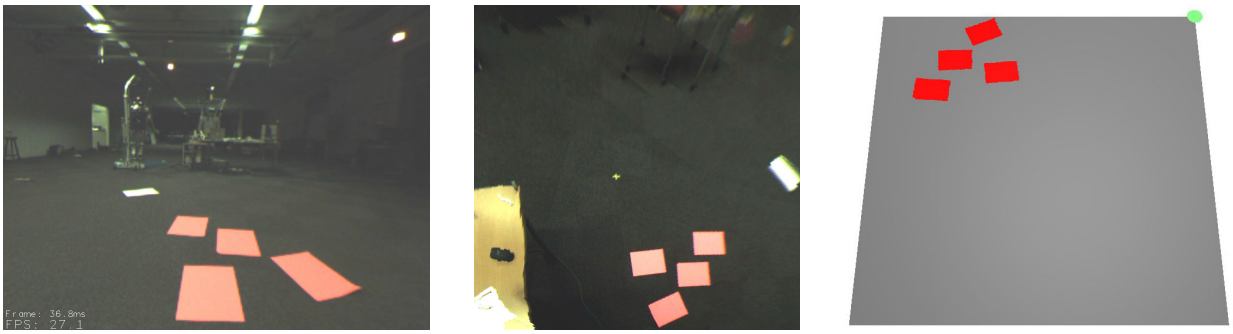
5 Evaluation of Sensing

The availability of ground truth positioning information from motion capture enables the precise localization of a variety of robot sensors such as cameras or range finders. Hence, we can build reliable global environment representations from sensor data, such as occupancy grids or height maps, upon which a robot navigation planner can operate. We compare these representations against ground truth by overlaying them onto projections of the real world. Using this comparison, we interactively develop and evaluate sensing algorithms and ensure their consistency with the physical robot surroundings.

The direct application of motion capture in localizing optical sensors is the construction of world models. A localized, calibrated camera can be used to calculate a 2D occupancy grid of the floor area. This is performed by means of video stream warping and segmentation. When the same method is applied to localize a range sensor, distance measurements can be converted into 2.5D height maps of the robot's surroundings. In both cases, maps are constructed by integrating sensor data accumulated over time as the robot moves through the environment. While each sensor measurement only reconstructs a part of the environment, as seen from a partial view of the scene, accurate global localization allows successive measurements to be co-registered in a global map of the robot environment. The resulting map can be used in planning trajectories for navigation and manipulation.

5.1 Reconstruction by Image Warping

Using images from a calibrated on or off-body camera, as shown in Figure 2, we reconstruct the robot's surroundings as if viewed from a virtual camera mounted overhead and observing the scene in its entirety. By accurately tracking the camera's position using motion capture, we are able to recover the full projection matrix. This enables a 2D collineation, or homography, between the floor and the image plane to be established, allowing incoming camera images to be warped onto the ground plane. A step of obstacle segmentation [23] then produces a 2D occupancy grid of the floor. Figure 3 gives an overview of the



(a) (b) (c)

Figure 4: Example camera image *(a)*. Synthesized ground plane view *(b)*. Corresponding environment map *(c)*.

reconstruction process.

For the purpose of building a 2D occupancy grid of the environment for biped navigation, we can assume that all scene points of interest lie in the $z = 0$ plane. Scene planarity then allows ground plane points $\mathbf{q} = (x, y, 1)^T$ in homogeneous coordinates to be related to points $\mathbf{p} = (u, v, 1)^T$ in the image plane via a 3×3 ground-image homography matrix $\mathcal{H}$ as $\mathbf{p} \equiv \mathcal{H}\mathbf{q}$. $\mathcal{H}$ can be constructed from the projection matrix $\mathcal{M}$ by considering the full camera projection equation

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ \mathbf{m}_1 & \mathbf{m}_2 & \mathbf{m}_3 & \mathbf{m}_4 \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{5}$$

and realizing that the constraint $z = 0$ cancels the contribution of column $\mathbf{m}_3$. $\mathcal{H}$ is thus simply composed of columns $\mathbf{m}_1$, $\mathbf{m}_2$ and $\mathbf{m}_4$, yielding the desired 3×3 planar homography, defined up to scale with 8 degrees of freedom.

The recovered homography matrix is square and hence easily inverted. $\mathcal{H}^{-1}$ can then be used to warp incoming camera images onto the ground plane and thus accumulate an output image, resembling a synthetic top-down view of the floor area. The ground plane image is thus incrementally constructed in real-time as the camera moves through the scene, selectively yielding information about the environment. Updates to the output image proceed by overwriting previously stored data. Figure 4(a) shows a camera image from a typical sequence and Figure 4(b) displays the corresponding synthesized floor view.

5.2 Reconstruction from Range Data

Using a marker-equipped CSEM SwissRanger SR-2 time-of-flight (TOF) range sensor [24], we are able to build 2.5D height maps of the environment containing arbitrary non-planar obstacles that the robot can step over, around, or onto during autonomous locomotion. Motion capture-based localization lets us convert range measurements into clouds of 3D points in world coordinates in real-time, from which environment height maps can be cumulatively constructed.

Offline we correct the raw range measurements with a per-pixel distance offset and estimate the sensor's focal length according to Zhang's camera model [25]. Per-pixel distances (u, v, d) are converted into camera-centric 3D coordinates $\mathbf{q}_c = (x, y, z)^T$ of the measured scene points. The extrinsic parameters $\mathcal{R}$ and $\mathbf{t}$ are recovered using motion capture. Combining the matrices, we construct the 4×4 transform matrix converting between the world frame and the camera frame in homogeneous coordinates:

$$\mathcal{T} = \begin{bmatrix} & \mathcal{R} & & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

$\mathcal{T}$ is easily inverted and can then be used to reconstruct each measured scene point in world coordinates via

$$\mathbf{q}_w \equiv \mathcal{T}^{-1}\mathbf{q}_c \tag{7}$$

The maximum z value of the measured scene points over a given position on the floor can then be recorded to build 2.5D height maps of the environment. Figure 5 shows an example reconstruction.

5.3 Registration with Ground Truth

Given a representation of the robot environment reconstructed by image warping or from range data,

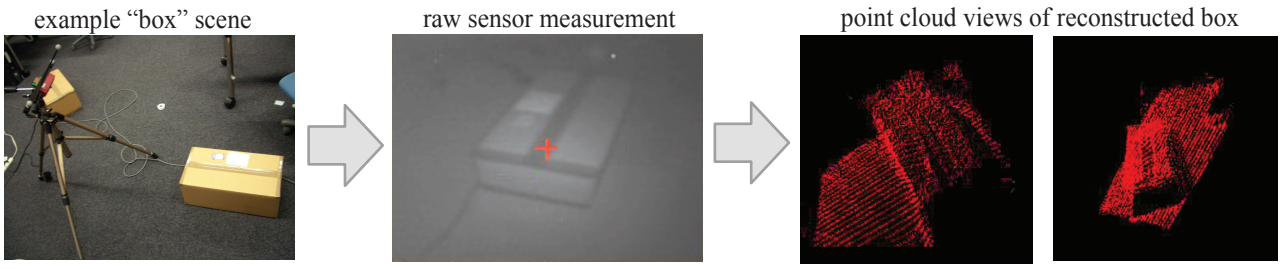
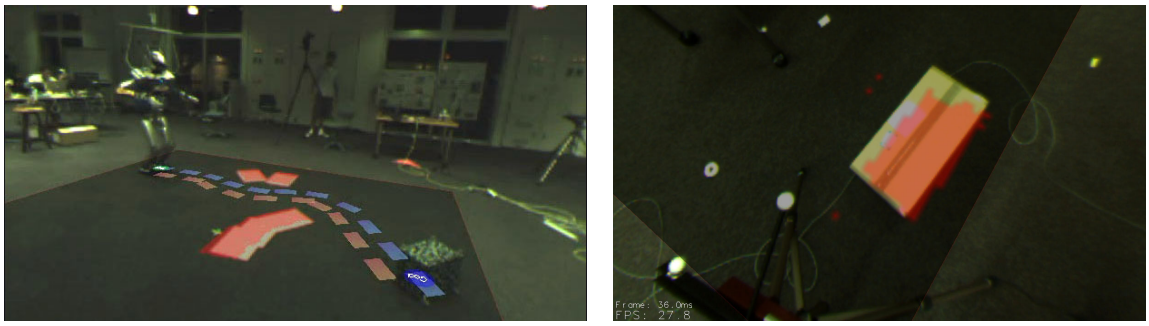


Figure 5: Swiss Ranger viewing an obstacle box placed on the floor *(left)*. Raw measurements recorded by the sensor *(center)*. Two views of the point cloud reconstruction of the box *(right)*.



(a) (b)

Figure 6: Environment reconstructions overlaid onto the world. (a) Occupancy grid generated from image-based reconstruction using the robot's camera. (b) planar projection of an obstacle recovered from range data.

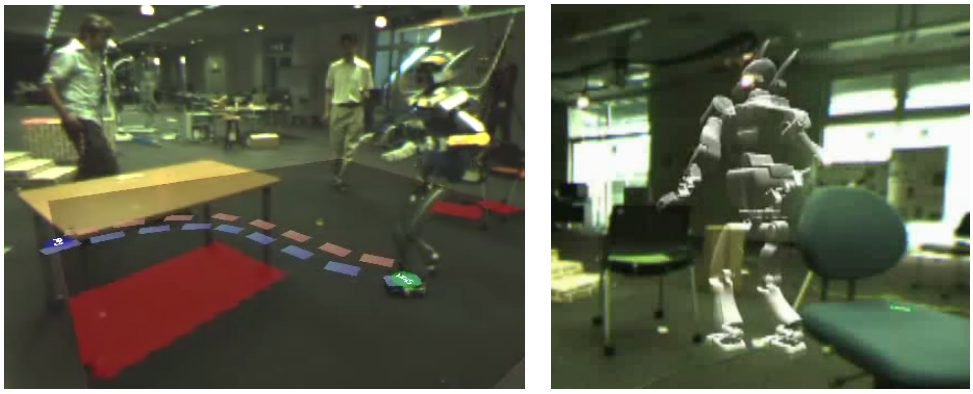
we can visually evaluate the accuracy of our perception algorithms and make parameter adjustments on-the-fly by overlaying the environment maps generated back onto a camera view of the scene. This enables us to verify that obstacles and free space in our environment reconstructions line up with their real-world counterparts, as illustrated in Figure 6.

6 Evaluation of Planning

Using the motion capture aided sensing systems described in the previous sections, we can control a robot to perform useful actions in real-world spaces. The various sensing methods can provide a range of data from near-perfect environment information to completely onboard vision-based sensing. These different approaches allow us to isolate the control system from errors introduced in the sensing, and then slowly bring in more realistic sensing once the planning and control algorithms have been validated. Unlike idealized sensing, we have no model of planning to which we can compare the robot's plans. However, through the use of the display techniques described in Section 4.3, we can expose the internal functionality of the planner. Using video overlay, we display diagnostic information about the planning and control process in physically relevant locations of the video stream.

We demonstrate this approach in the realm of biped navigation planning. In this task, we have a humanoid robot, HRP-2, in a real-world environment with several obstacles. The robot must plan a safe sequence of actions to convey itself from its current configuration to some goal location. In these experiments, the goal and obstacles were moved while the robot was walking, requiring the robot to constantly update its plan to account for the new information. The planning algorithm itself evaluates candidate footstep locations, allowing it to find a sequence of footholds that can carry it through a cluttered environment [20].

There are three levels of varying reality for testing this system:



(a) (b)

Figure 7: Augmenting reality for visualization of planning and execution. (a) Footstep plan displayed onto the world. (b) Augmented reality with a simulated robot amongst real obstacles.

- Motion capture obstacle recognition: As described in Section 4.2, we can reconstruct the geometry of various objects in our environment. This allows us to build obstacle maps by projecting these objects to the ground plane or into a height map. Figure 7(a) shows an example of one of these experiments.
- Localized sensors: Section 5 describes the process by which motion capture data can be used to localize sensors for building maps of the environment. We can test using onboard sensing and real vision data, but with global registration performed by the motion capture system. An example of an experiment performed with this data is shown in Figure 6(a).
- Self-contained vision: When the motion capture data are removed completely, the robot must use its own physical and visual odometry to build maps of the environment.

6.1 Visual Projection: Footstep Plans

Figures 6(a) and 7(a) show examples of control system visualization during online robot experiments. In these cases, the system has planned out the sequence of footsteps it wishes to take to reach some goal configuration. For each step, it has computed the 3D position and orientation of the foot. Through the use of augmented reality, the planned footsteps can be overlaid in real-time onto the environment. The red and blue rectangles represent the steps for the right and left feet that the robot intends to take. This path is constantly updated as the robot replans while walking. This display helps expose the planning process to identify errors and gain insight into the performance of the algorithm.

6.2 Temporal Projection: Virtual Robot

One of the components of our overall system that we would like to replace for testing purposes is the robot itself. One solution to is to build a simulated environment for experimentation. However, we would like to continue to use the real world as much as possible, rather than using a completely fabricated environment. Within our framework, we can continue to use real-world obstacles and sensors, and merely replace the robot with a simulated avatar. Figure 7(b) shows the augmented reality of our simulated robot traversing a real environment. Note that for this navigation task, the robot is not manipulating the environment. The obstacles themselves can be moved during the experiments, but we do not need to close the loop on robotic manipulation.

6.3 Objects and the Robot's Perception

In addition to complete replacement of all sensing with perfect ground truth data, we can simulate varying degrees of realistic sensors. We can slowly increase the realism of the data which the system must handle. This approach can isolate specific sources of error, and determine to which the the control system is most sensitive. For example, by knowing the locations and positions of all objects as well as the robot's sensors, we can determine which objects are detectable by the robot at any given point in time.

Hence, simulated sensors can be implemented with realistic limits and coverage.

7 Evaluation of Control

Having perceived the scene and constructed plans for navigation and manipulation, the remaining aspect of testing is the control of the robot. Our objective in terms of control testing is to maximize the safety of the robot and the environment. To accomplish this, we perform hardware in the loop simulations while gradually introducing real components.

7.1 Virtual Objects

In simulation, it is possible to analyze the interaction of a robot with a virtual object by constructing a geometric and dynamic model of the object. Our system performs identical operations during on-line experiments. Suppose we introduce geometric virtual obstacles into the lab space. Overlaying these obstacles onto the view of a tracked camera allows us to perceive them as though they were part of the scene.

During a navigation task, the robot treats virtual obstacles as real ones and constructs plans to walk around them. To execute these plans, the robot computes dynamically stable walking patterns that satisfy ZMP conditions [26]. The robot performs active balance compensation using foot force sensors. While evaluating the performance of our controller, we can use the overlay view to ensure that the robot satisfies the constraints of our environment. In case of a failure, we observe and detect virtual collisions without affecting the robot hardware.

Similarly, these concepts can be applied towards grasping and manipulation. When the robot grasps a virtual object, we can simulate the presence and geometry of this object. Furthermore, since the world model is directly given to the robot, optical information is not required. Consequently, we can instrument our environment in any desired way to gauge the interaction with a virtual object without a physical presence.

7.2 Precise Localization

Having established the basic validity of our controllers we continue by introducing actual objects such as tables and chairs into the scene. Particularly during manipulation, we are interested in the ability of our robot to correctly calculate inverse kinematics and perform force control on an object during physical interaction.

Generally, this sort of experimentation would require either fixing the initial conditions of the robot and environment, or asking the the robot to sense and acquire a world model prior to every experiment. The hybrid experimental model avoids the rigidity of the former approach and the overhead time required for the latter. As we described in Section 4.2, employing the motion capture system as an idealized sensor provides a real-time feed of the virtual world geometry that corresponds to the robot's actual surroundings.

Using the idealized optical sensor, we can focus our efforts directly on algorithms for making contact with the object and evaluating the higher frequency feedback required for force control. In later testing, we can slowly introduce the errors from a local perception model and safely continue the stabilization process.

7.3 Gantry Control

The final concern in testing a humanoid robot is the lack of static stability that is inherent in dynamic walking. During any task of locomotion or manipulation, a humanoid robot is at risk of falling. Typically, a small gantry is used to closely follow and secure the robot. However, the physical presence of the gantry and its operator prevent us from testing fine manipulation or navigation that requires the close proximity of objects.

To bypass this problem, our laboratory implements a ceiling suspended gantry that can follow the robot throughout the experimental space. Having acquired the absolute positioning of the robot from motion capture, this gantry is PD controlled to follow the robot as it autonomously explores the space.

This final component not only lets us to test the robot in arbitrary cluttered environments, but also enables experiments that typically require four or five operators to be safely performed by a single researcher.

8 Discussion

We have presented a novel experimental paradigm that leverages the recent advances in optical motion capture speed and accuracy to enable simultaneous online testing of complex robotic system components in a hybrid real-virtual world. We believe that this new approach enabled us to achieve rapid development and validation testing on each of the perception,

planning, and control subsystems of our autonomous humanoid robot platform. We hope that this powerful combination of vision technology and robotics development will lead to faster realization of complex autonomous systems with a high degree of reliability.

Future work includes the investigation of automated methods for environment modeling. Ideally, an object with markers could be moved through the environment and immediately modeled for application in the hybrid simulation. Machine learning and optimization techniques should be explored for automatic sensor calibration in the context of a ground truth world model. The visualizations can be enhanced by fusing local sensing such as gyroscopes and force sensors into the virtual environment. While motion capture largely reflects the kinematics of the scene, local sensors could enhance the accuracy of information on dynamics.

References

[1] F. Kanehiro et. al. Open architecture humanoid robotics plantform. In *IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 24–30, 2002.

[2] J.J. Kuffner, S. Kagami, M. Inaba, and H. Inoue. Graphical simulation and high-level control of humanoid robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'00)*, 2000.

[3] O. Khatib, O. Brock, K-S Chang, F. Conti, D. Ruspini, and L. Sentis. Robotics and interactive simulation. *Communications of the ACM*, 45(3):46–51, 2002.

[4] David Baraff. Analytical methods for dynamic simulation of non-penetrating rigid bodies. *Computer Graphics*, 23(3):223–232, 1989.

[5] Brian V. Mirtich. *Impulse-base Dynamic Simulation of Rigid Body Systems*. PhD thesis, Dept. of Computer Science, University of California at Berkeley, 1996.

[6] D. Ruspini and O. Khatib. A framework for multi-contact multi-body dynamic simulation and haptic display, 2000.

[7] K. Yamane and Y. Nakamura. Efficient parallel dynamics computation of human figures. In *IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 530–537, 2002.

[8] F. Kanehiro, H. Hirukawa, and S. Kajita. Openhrp: Open architecture humanoid robotics platform. *Int. J. of Robotics Research*, 23(2):155, 2004.

[9] J. de Carufel, E. Martin, and J. Piedboeuf. Control strategies for hardware-in-the-loop simulation of flexible space robots. In *In Proc. Control Theory and Applications*, volume 147, 2000.

[10] O. Lorch et. al. Vigwam: An emulation environment for a vision guided virtual walking machine. In *Proc. IEEE Int. Conf. on Humanoid Robotics (Humanoids'00)*, 2000.

[11] R. T. Azuma. A survey of augmented reality. *Teleoperators and Virtual Environments*, 6(4):355–385, 1997.

[12] R. Azuma, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. A survey of augmented reality. *Computer Graphics and Applications, IEEE*, 21(6):34–47, 2001.

[13] P. Milgram and J. Ballantyne. Real world teleoperation via virtual environment modeling. In *Int. Conf. on Artificial Reality & Tele-existence ICAT97*, 1997.

[14] K. Kutulakos and J. Vallino. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):1–20, 1998.

[15] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. *Int. J. of Computers in Biology and Medicine*, 25(2):249–260, 1995.

[16] K. Dorfmuller. Robust tracking for augmented reality using retroreflective markers. *Computers and Graphics*, 23(6):795–800, 1999.

[17] Y. Argotti, L. Davis, V. Outters, and J. Rolland. Dynamic superimposition of synthetic objects on rigid and simple-deformable real objects. *Computers and Graphics*, 26(6):919, 2002.

[18] A. State, G. Hirota, D.T. Chen, W.F. Garrett, and M.A. Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In *In Proc. SIGGRAPH'96*, page 429, 1996.

[19] Motion Analysis Corporation. Motion Analysis Eagle Digital System. Web: `http://www.motionanalysis.com`.

[20] Joel Chestnutt, James Kuffner, Koichi Nishiwaki, and Satoshi Kagami. Planning biped navigation strategies in complex environments. In *Proc. IEEE Int. Conf. on Humanoid Robotics*, Karlsruhe, Germany, October 2003.

[21] M. Stilman and J.J. Kuffner. Navigation among movable obstacles: Real-time reasoning in complex environments. In *Proc. IEEE Int. Conf. on Humanoid Robotics (Humanoids'04)*, 2004.

[22] R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 364–374, Miami Beach, FL, 1986.

[23] James Bruce, Tucker Balch, and Manuela Veloso. Fast and inexpensive color image segmentation for interactive robots. In *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS-2000)*, Japan, October 2000.

[24] Thierry Oggier, Michael Lehmann, Rolf Kaufmann, Matthias Schweizer, Michael Richter, Peter Metzler, Graham Lang, Felix Lustenberger, and Nicolas Blanc. An all-solid-state optical range camera for 3D real-time imaging with sub-centimeter depth resolution (SwissRanger). Available online at: `http://www.swissranger.ch`, 2004.

[25] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proc. of the Int. Conf. on Computer Vision (ICCV '99)*, pages 666–673, Corfu, Greece, September 1999.

[26] Koichi Nishiwaki, Satoshi Kagami, Yasuo Kuniyoshi, Masayuki Inaba, and Hirochika Inoue. Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired zmp. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2684–2689, 2002.