The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Predictive Display for Mobile Manipulators in Unknown Environments Using Online Vision-based Monocular Modeling and Localization

David Lovi, Neil Birkbeck, Alejandro Hernandez Herdocia, Adam Rachmielowski,
Martin Jägersand, Dana Cobzaş
University of Alberta, Deptartment of Computing Science

Abstract—

To tele-operate a robot, visual feedback is critical. However, communication channel latency can delay feedback to the point where the operator is impeded in performing his task. This work presents a vision-based "predictive display" system that compensates for visual delay. The approach is online and relatively uncalibrated, thus it has the advantage of being useful in unknown environments and many applications. From monocular eye-in-hand video, we incrementally compute a 3D graphics model of the robot site in real time using our new technique. The method exploits free-space/occlusion constraints on the scene to produce a physically consistent mesh. Novel vantage points are immediately rendered in response to the operator's control commands, without waiting for delayed video. We implement a full prototype tele-operation system where the operator controls, via a PHANTOM Omni device, a Barrett WAM robot mounted on a mobile Segway. Experiments with this setup validate the efficacy of the proposed approach. We demonstrate significant improvement in task completion time with predictive display on a real robot, while our previous related results were established only in simulation.

I. Introduction

In tele-robotics, a human operator controls a remotely located robot. The operator sends control commands to the robot and receives sensory feedback via a communication channel. Such remote tele-operation lends itself to many applications where it would be dangerous or impractical for a human to perform the robot's task directly. Space and marine robotics, tele-surgery, and remote bomb defusal are just a few examples.

Communication delays are a fundamental problem. When the operator sends commands to the robot, he expects sensory feedback to reflect his inputs in a natural and transparent way. This is essential for closing the master-slave control loop. However, in practice sensory feedback is delayed by the round-trip latency of the communication channel. Therefore, simple streaming of video and haptic information typically proves insufficient. Visual delays as small as 0.3 seconds negatively impact operator performance and produce a sense of decoupling the human's motions from the robot, resulting in effectively open-loop "move and wait" control [1], [2], [3], [4], [5]. This magnitude of delay is common across the internet, and *e.g.* ground-to-space tele-operation suffers unavoidably larger delay due to the finite speed of light.

The focus of this paper is to ameliorate the effects of visual delay via predictive display. Predictive display refers to rendering a visualization of the robot site directly in response to the operator's control commands, without waiting for delayed video.

Our system accomplishes this via online computer-vision techniques at the robot site; from video, a 3D model is acquired and updated in real time and sent back to the operator for immediate visualization. The operator's rendering pose is determined from his control commands and forward kinematics, and from monocular localization of the robot's mobile base w.r.t. the world. Specifically, this paper presents three main contributions.

- An online vision subsystem comprised of a synthesis of our recent free-space carving technique [6] and PTAM [7], providing robust localization and online model construction for photorealistic predictive display.
- Integration into a full prototype tele-robotics system with a Barrett WAM arm and Segway mobile base.
- Experiments that validate and demonstrate the effectiveness of this predictive display approach.

Traditionally, predictive display has been accomplished in highly precalibrated settings by superimposing hand-modeled wireframe and solid-model overlays of the robot manipulator and scene objects atop delayed video [1], [8], [9]. This approach supposes a known environment and non-moving external camera. More recent work aims to produce photorealistic predictive display in less calibrated scenarios [10], [11], [12], [13]. Our work falls into this category; we provide for an eye-in-hand moving camera and do not require a 3D model a priori.

Image-based rendering techniques have been applied to photorealistic predictive display, including pure image-based synthesis [11] and hybrid geometric approaches [14], [12]. To date however, these techniques require an offline reconstruction step or a lengthy online learning phase to get a sufficiently dense image-sampling. Burkert *et al.* describe an online depth-fusion technique for predictive display that acquires a 3D model using a stereo camera rig [10]. However, their system takes upwards of 30 s to integrate each new depth map, and it exhausts computational resources, both CPU and GPU. Additionally, their implementation lacks online localization of the camera rig. In contrast to these systems, we demonstrate real-time tracking, localization, and 3D scene reconstruction using a single mobile laptop CPU.

Rachmielowski's system is closely related to ours [13]; we build upon this work. His system reconstructs a sparse camera-set and 3D point structure using online SLAM. It

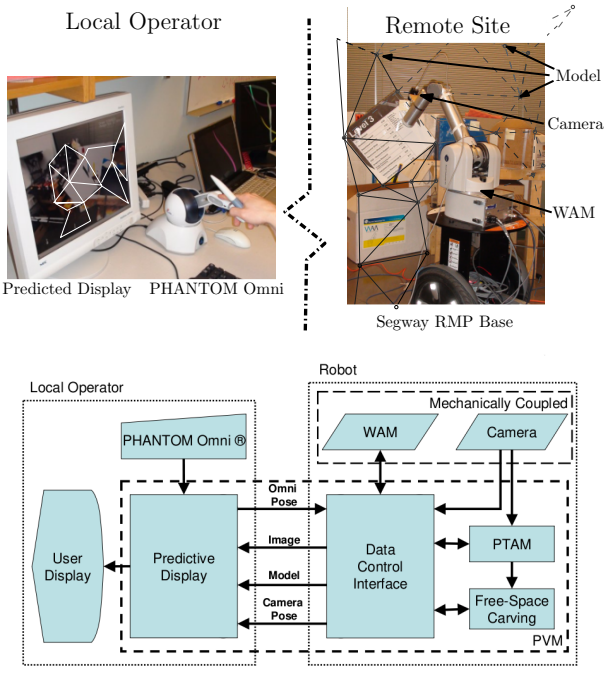


Fig. 1. Top: the operator tele-operates the robot from the local site; the model is computed at the remote site and transferred to the operator for predictive display. Bottom: system components and data flow.

creates a coarse view-dependent geometry by connecting projected 3D points and then backprojecting the 2D mesh into a 3D model. Predictive rendering is achieved by projective texturing from keyframes. This method enjoys similar benefits to our own: online performance, support for a moving camera without rigorous precalibration, no need for a 3D model a priori, and applicability to a variety of unknown environments. However, the integration with a real robot was preliminary, and thus experimental validation in [13] was performed primarily in simulation.

In contrast, our system offers improvements. Instead of using a view-dependent rendering proxy, we infer a single coherent view-consistent proxy that respects physical free-space/visibility constraints on the scene. Moreover, we provide experimental evaluation on a real Barrett WAM robot.

II. Systems

Our system consists of a WAM robot mounted on a Segway RMP. A single camera is mounted on the end-effector of the WAM. We use a PHANTOM Omni haptic device to tele-operate the WAM through a joint angle mapping; see Fig. 1.

The operator is presented with immediate visual feedback of his motions. This predictive display utilizes a coarse geometry obtained from our free-space carving technique that executes in parallel with a vision-based tracking and mapping process at the robot site. The vision process tracks and maps independent of any knowledge of the robot motion. On the operator site, the received geometric proxy is rendered using the *predicted* camera pose and textures from delayed images. The predicted pose of the camera corresponds to the location where the robot-mounted camera will be when the robot receives the joint angles for the current Omni pose.

Currently, the Segway RMP is not actuated; the operator has only direct control of the kinematic arm. However, our system supports correct predictive display even in the presence of unknown motion of the platform. As a consequence of our vision-process being uncoupled from the robot, we use discrepancies between our kinematic computed pose and the vision-process tracked pose to determine the relative motion of the base.

The remainder of this section outlines the modules of our system. Specifically, first we describe the vision-based 3D modeling algorithms running at the robot site (Section II-A). Then we discuss how this model, delayed images, and remote camera pose are used in predictive display (Section II-B). As the vision-process is running independently, the reconstructed model must first be registered online in the robot coordinate system (Section II-C). Finally, we discuss our particular choice of joint angle mapping from the PHANTOM Omni to the WAM (Section II-D).

A. Vision Modeling by Free-Space Volume Carving

For the vision module, we run two components at the robot site: incremental free-space carving and PTAM; see Fig. 1.

The carving module constructs a 3D graphics model of the robot site. The approach is volumetric: we chisel away sections of space that violate free-space constraints on the scene. When a camera images a 3D patch of scene surface, we know that the volume comprised of the rays of projection from the patch to the camera's optic center must be vacant, and therefore consist of "free space." Fig. 2(a) illustrates.

In practice, instead of generalized patches, our carving module takes as input a sampled point cloud of 3D reconstructed world features $\{P\}$. It also takes a history of keyframe camera poses with optic centers $\{O\}$, as well as visibility information $\{\overline{OP}\}$ that relates which points in $\{P\}$ are visible from each view of $\{O\}$. See Fig. 2(b).

Such inputs are readily available from any SLAM-like tracking system operating on the robot-camera's video stream. We obtain them from PTAM [7]. PTAM additionally computes at each time t the current localized camera pose w.r.t. the world, $\mathbf{E}_p^t$. We use this pose to determine the robot's base position in real-time; see Section II-C. PTAM was chosen as our base tracking system because it is robust and effective; it produces a dense and accurate point cloud, and it reliably relocalizes to recover from tracking failures.

To carve, we adaptively discretize space via the 3D Delaunay triangulation of the input point set $\{P\}$. This partitions space into a connected set of tetrahedra that span $\{P\}$'s convex hull. The method then marks the tetrahedra that intersect any free-space constraint $\overline{OP}$ as empty. In this way, a physically consistent carved scene model is produced.

As PTAM acquires more images, our inputs $\{O\}$, $\{P\}$, and $\{\overline{OP}\}$ continuously change online. The free-space carving method incrementally reconciles such changes and expands the model. Modifications to $\{P\}$, be it from the addition of new points or from outlier deletion, entail rediscretiza-

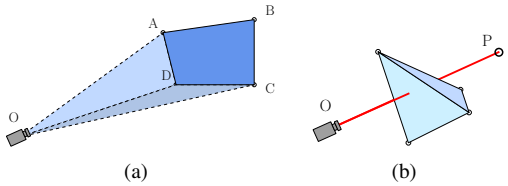


Fig. 2. Free-space constraints. (a) The general concept. A camera O observes a surface patch, here the quadrilateral $ABCD$. The pyramidal volume $ABCDO$ must be empty; otherwise, the patch would be occluded. (b) Our chosen representation of free-space constraints. The carving method considers only points P instead of generalized patches. Therefore our free-space constraints are infinitesimally thin volumes, the line segments, $\overline{OP}$.

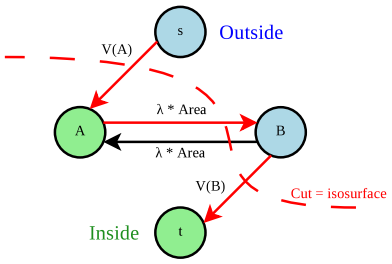


Fig. 4. A graph with two tetrahedra A and B. Edges between A and B carry the regularization term; edges connecting to s and t represent the data term. (V denotes volume). The cut's cost is the sum of edge weights leading from s's connected component to t's (red edges).

tion of a subset of space. To support real-time reconstruction, we therefore associate with each tetrahedron the set of free-space constraints that intersect it. Using this mapping, we efficiently determine the minimal set of constraints that must be reprocessed in response to an incremental change in the inputs. See Figure 3 for an illustration of the algorithm. Other types of changes to the inputs (*e.g.* bundle adjustment) are handled analogously. To ensure the real-time quality of our system, we employ a simple heuristic: in each tetrahedron's constraint set, we retain only the K most dissimilar constraints, where K is a fixed parameter. For full algorithmic details and run-time complexity analysis, refer to [6].

Given a carving, we still need to compute the isosurface as a conventional 3D graphics mesh. Because tetrahedral facets are triangles, this can be straightforwardly computed as the set of facets that border adjacent tetrahedra with differing labels ("carved" or "uncarved").

We implement an improvement over our previous work [6] by regularizing the isosurface computation. This results in smoother meshes that are better rendering proxies. Let $\mathbf{x}$ denote a 3D point $(x, y, z)^T$, and let $\triangle(\mathbf{x})$ be the tetrahedron containing $\mathbf{x}$. Let $u(\mathbf{x}) \in \{0, 1\}$ be a binary labeling denoting whether $\mathbf{x}$ is carved or uncarved, and let $v(\triangle(\mathbf{x}))$ be the labeling for tetrahedron $\triangle(\mathbf{x})$ provided by our base algorithm. In essence, we minimize the following (continuous) energy functional, but over a discrete carving $u(\triangle)$:

$$\iiint |u(\mathbf{x}) - v(\triangle(\mathbf{x}))|\, d\mathbf{x} + \lambda \iiint H(\|\nabla u\|)\, d\mathbf{x}. \quad (1)$$

Here H is the Heaviside step function defined such that $H(0) = 0$, and λ is a scalar regularization parameter. (λ was set to 0.75 in all of our experiments.) The left-hand integral is the data term, and the right-hand integral evaluates to the surface area of the isosurface of u.

Put another way, we find the optimal carving $u(\triangle)$ that minimizes the volume that disagrees with the original carving $v(\triangle)$ plus a surface-area penalty term. We then extract the isosurface of u.

To optimize, we cast the minimization as a discrete graph-cut problem. We construct the graph as follows. The vertices correspond one-to-one with each tetrahedron, except for an additional source s and sink t. Node s is associated to the label 0 = carved, and t to 1 = uncarved. To encode the data term, each node representing a tetrahedron $\triangle$ with original label $v(\triangle) = 0$ connects to s by an edge with weight equal to $\triangle$'s volume. Nodes with the opposite label connect similarly to t. To encode the penalty term, nodes corresponding to adjacent tetrahedra connect to each other by directed edges with weight λA, where A is the area of their shared facet. See Fig. 4 for an illustration.

Because the tetrahedra are four-connected, our graph has a topology common to many graph-cut problems in computer vision. To compute the regularized carving u, we use an efficient algorithm optimized for such graphs [15].

Typically, the full graph-cut regularization and isosurface extraction runs in about 0.25 seconds on a single core of an Intel Core2 Duo CPU T5550 @ 1.83GHz. In practice, to allow other threads and processes to run, we restrict the rate at which we update our isosurface model to 1 Hz.

B. Display

In order to correctly predict what the robot camera will observe when the operator moves the Omni, the predictive display module must have access to the following quantities:

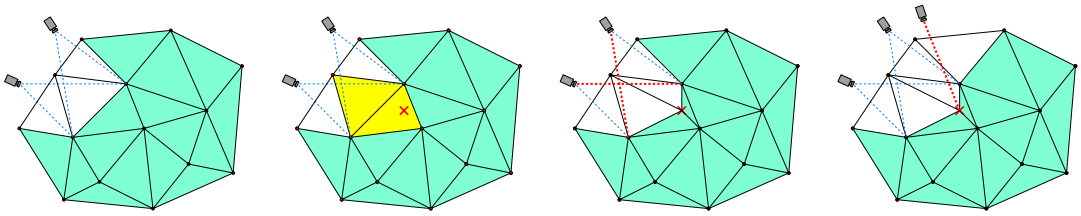
- A set of images with known camera pose
- A geometric model
- A predicted camera pose

The geometric model can then be textured with one or more of the saved input images, and a novel image can be rendered from the predicted camera pose.

As indicated in Fig. 1, the images, corresponding camera pose, and geometric model come (delayed) from the remote site. The predicted camera pose must take into account the mapping from the input device to the robot (discussed in detail in Section II-D). In our setup, the input device controls only the WAM arm. Therefore, in the case of a fixed platform, the predicted camera pose can be obtained through a forward kinematic mapping of the current joint angles from the input device to the WAM arm. The pose of the camera relative to the end-effector can easily be pre-calibrated.

Incorporating a mobile base into the predicted camera pose is straightforward provided the mobile base pose is known (Section II-C). In such cases, the predicted pose can be obtained by the forward kinematic mapping, as before, concatenated with the base transformation.

Although more elaborate schemes are possible, we have used only the most recent image and its camera pose to



(a) Initial (b) Conflicts (c) Retriangulation (d) Final Carving

Fig. 3. A 2D illustration of incremental free-space carving. (a) The initial triangulation. The blue dashed lines are free-space constraints currently in the triangulation. Shaded cells have not yet been carved. (b) An incoming point (red cross). The yellow cells are in Delaunay-conflict with the point because it falls inside their circumcircles. (c) The yellow cells were deleted and rediscretized. The red free-space constraints (bolded) belonged to the deleted tetrahedra; they are now used to carve away two of the four new tetrahedra. (d) Finally, the new free-space constraint(s) from the current view are applied.

texture the most recent geometric model.

C. Calibration

A field robot cannot rely on a specific calibration of the scene relative to the robot. Upon initialization, we align the structure from our vision process into the robot's coordinate frame. Subsequent pose information from the vision process is then used to estimate the base motion of the robot. An important consequence of the system not using odometry is that our updated base transformations can allow for arbitrary motion (*e.g.*, if the robot is on a ship moving in waves and is modeling and tracking an adjacent ship, the base motion of the robot will be compensated as long as the visual tracking is reliable).

PTAM provides an estimate of the camera in the world, and our kinematics give an estimate of the camera relative to the base; discrepancies between the two coordinates imply motion in the base. We first discuss how PTAM's coordinates can be aligned to our initial kinematic coordinate frame, and then we present a simple filtering approach to incorporate changes in PTAM's camera pose into the estimated base motion of our platform.

1) Relative Calibration: Let $\mathbf{E}_m^i$ and $\mathbf{E}_p^i$ denote the pose of the camera for the model (subscript m) and PTAM (subscript p) in their respective frames at time i:

$$\mathbf{E}_m^i = \begin{bmatrix} \mathbf{R}_m^i & \mathbf{t}_m^i \\ \mathbf{0} & 1 \end{bmatrix} \tag{2}$$

$$\mathbf{E}_p^i = \begin{bmatrix} \mathbf{R}_p^i & \mathbf{t}_p^i \\ \mathbf{0} & 1 \end{bmatrix} \tag{3}$$

The change of coordinates is a metric transformation, ${}^m\mathbf{W}_p$, that aligns PTAM coordinates with our kinematic coordinates, $X_m = {}^m\mathbf{W}_p X_p$, and can be simply obtained from two corresponding camera poses:

$${}^m\mathbf{W}_p = {}^m\hat{\mathbf{W}}_p diag(s,s,s,1) \tag{4}$$

$${}^m\hat{\mathbf{W}}_p = \mathbf{E}_m^1 \begin{bmatrix} \mathbf{R}_p^1 & s\mathbf{t}_p^1 \\ \mathbf{0} & 1 \end{bmatrix}^{-1} \tag{5}$$

$$s = \frac{|\mathbf{t}_m^1 - \mathbf{t}_m^0|}{|\mathbf{t}_p^1 - \mathbf{t}_p^0|} \tag{6}$$

The transformation aligns the coordinates for $\mathbf{E}_m^1$ and only uses the correspondences asymmetrically. To utilize both the correspondences equally, we refine the solution by minimizing reprojection error. Given at least two corresponding coordinate frames and a set of 3D points in PTAM's coordinates $\{\mathbf{X}_p^k\}$, the reprojection error is then

$$\sum_{i,k} |\Pi(\mathbf{K}\mathbb{I}({}^p\hat{\mathbf{W}}_m \mathbf{E}_m^i)^{-1}(s\mathbf{X}_p^k)) - \Pi(\mathbf{K}\mathbb{I}(\mathbf{E}_p^i)^{-1}\mathbf{X}_p^k)|^2$$

where Π is the perspective division operator, $\mathbb{I}$ is the 3x4 identity matrix, $\mathbf{K}$ is the $3x3$ matrix of internal camera parameters, and ${}^p\hat{\mathbf{W}}_m = ({}^m\hat{\mathbf{W}}_p)^{-1}$.

This formulation is similar to resection in computer vision, with the exception that the transformation is common to all the correspondences. Furthermore, notice that no measured image correspondences are used; the reprojection error measures the difference between the 3D points from PTAM and their projection in the transformed kinematic coordinates. That is, the 3D points are used as representative samples of the scene.

If necessary, the pose of the camera on the kinematic structure can be refined by adding an additional camera update transformation, $\mathbf{T}_{cam}$, in the objective:

$$\sum_{i,k} |\Pi(\mathbf{K}\mathbb{I}({}^p\hat{\mathbf{W}}_m \mathbf{E}_m^i \mathbf{T}_{cam})^{-1}(s\mathbf{X}_p^k)) - \Pi(\mathbf{K}\mathbb{I}(\mathbf{E}_p^i)^{-1}\mathbf{X}_p^k)|^2$$

Optimization of this objective is only necessary once to determine the pose of the camera on the arm.

2) Base Motion: After alignment of the coordinate frames, any discrepancies from PTAM's transformed camera pose must be either due to mistrack or mobile platform motion. As PTAM provides an accurate measure of the tracking status, we rule out mistracking and can estimate the mobile platform motion as follows:

$$\hat{\mathbf{B}}(i)\mathbf{E}_m^i = {}^m\hat{\mathbf{W}}_p \begin{bmatrix} \mathbf{R}_p^i & s\mathbf{t}_p^i \\ \mathbf{0} & 1 \end{bmatrix} \tag{7}$$

$$\hat{\mathbf{B}}(i) = {}^m\hat{\mathbf{W}}_p \begin{bmatrix} \mathbf{R}_p^i & s\mathbf{t}_p^i \\ \mathbf{0} & 1 \end{bmatrix} (\mathbf{E}_m^i)^{-1} \tag{8}$$

Our filtered estimate, is then:

$$\mathbf{B}(t) = (1-\lambda)\hat{\mathbf{B}}(t) + \lambda\mathbf{B}(t-1)$$

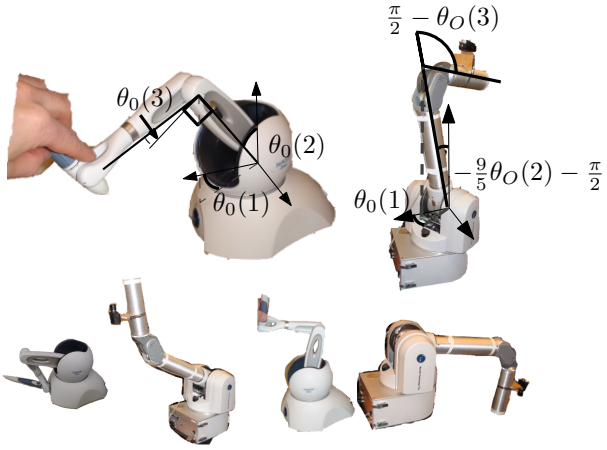


Fig. 5. The mapping between the PHANTOM Omni and the WAM robot.

where, in practice, we perform the above interpolation using SLERP for the quaternion component and linear interpolation on the translation. For all of our experiments $\lambda = 0.95$.

D. Input Device Mapping to WAM Robot

Our WAM robot is 4 DOF. It has shoulder and elbow joints, but no wrist. These freedoms do not directly agree with the first freedoms of the PHANTOM, therefore we use the mapping illustrated in Fig. 5. In this mapping, the roll around the first WAM link (the third joint) is fixed to zero. The rotation of the PHANTOM base directly maps to the base rotation of the WAM. The angles from the first link are scaled to make up for the small range of the PHANTOM, and the last link is mapped with an offset. Letting θ_O be the OMNI angles, the mapping is

$$\theta_{WAM} = [\theta_O(1), -\frac{9}{5}\theta_O(2) - \frac{\pi}{2}, 0, \frac{\pi}{2} - \theta_O(3)]$$

These joint angles are forwarded from the operator site through the PVM communication channel to the remote site at 30 Hz. The WAM robot control loop runs at 500 Hz and uses joint PID controllers to follow smooth trajectories derived from the incoming angles. The smooth trajectories have a truncated trapezoid velocity profile with a maximum velocity and maximum acceleration of 1 rad/s and 1 rad/s^2. Upon receiving a new command, both the current velocity and joint angles are used to fit the trapezoid shaped velocity. This ensures that abrupt and possibly damaging motions by the operator are not translated directly to the robot.

III. Experiments

In this section we present three experiments performed with the system. The main purpose is to test the volume carving technique and to validate the predictive display system in a real robotics setup.

A. Mobile Base Motion

In this experiment, the robot's base is arbitrarily moved away from the initial position by a lab-mate pulling it, like shown in the supplemental video. The purpose is to test the model building from an unknown environment, and to show that it is possible to move or perturb the robot's position while the predictive display appropriately compensates.

The experiment was run by starting the system in an arbitrary location in a room and initializing the model in a local region. After the system has enough features to build a model, the base is moved to a new position, while visual features are tracked to update the base-to-model location[1]. The model continues to be built by adding newly carved space. Using this method, we built a model of our robotics laboratory. Fig. 6 shows the reconstruction of the room seen from various vantage points.

B. Alignment Task

Alignment tasks are common in manipulation[2]. In this case, we used visual targets (letters) in the scene which the user had to align with the rendered reference in the video display. When the rendered target matches the real one then the alignment is satisfied. This experiment evaluates and tests alignment performance under predictive display. We compare three modes of visual feedback: non-delayed video, 0.3 s delayed video, and predictive display with 0.3 s delay.

This experiment is largely inspired by and similar to the one presented by Rachmielowski [13], yet ours is conducted on a real-world robot instead of a simulated graphics environment. To be able to compare the timings of several subjects, in this experiment the 3D model was acquired once by the vision system, and the same model was used for all subjects. Texturing used video from the robot camera, and this varied for each trial.

The experiment was conducted first by running a warmup where the user familiarizes himself with the input device and kinematics of the robot, as well as the three visual modes. After the warmup session, the timed experiment starts. Each user performs a total of three trials in each of the three modes. For each trial, the display mode and scene configuration are drawn randomly without replacement. The scene is comprised of four targets each placed in one out of six possible calibrated positions. The user has to first align A, then B, then C and finally D. An alignment is satisfied when the user places the robot in a position which is close enough[3] to the desired position.

The user is only allowed to look at the display and not at the scene where the robot is operating. Fig. 7 shows an alignment task where the user is controlling the robot to align the rendered A with the actual A in the scene.

Due to the time needed to arrange the physical scene configuration, the experiments involved only 36 alignments by each of five lab colleagues for a total of 180 alignments. Despite the small number of test subjects, the results were consistent with the 1200 alignments on a simulated graphics scene [13]. This experiment illustrates that the

[1]The range of the displacement of the mobile base is only limited by the fact that enough features have to remain trackable at all times

[2]For example, in operations like putting a wrench on a bolt

[3]A threshold is set on the distance from the exact alignment to the actual alignment, as well as a 500 ms dwelling time. The threshold was tuned to be reasonably challenging: satisfiable, but near the limit of human precision using a PHANTOM Omni and the 640×480 video / texture feed resolution.

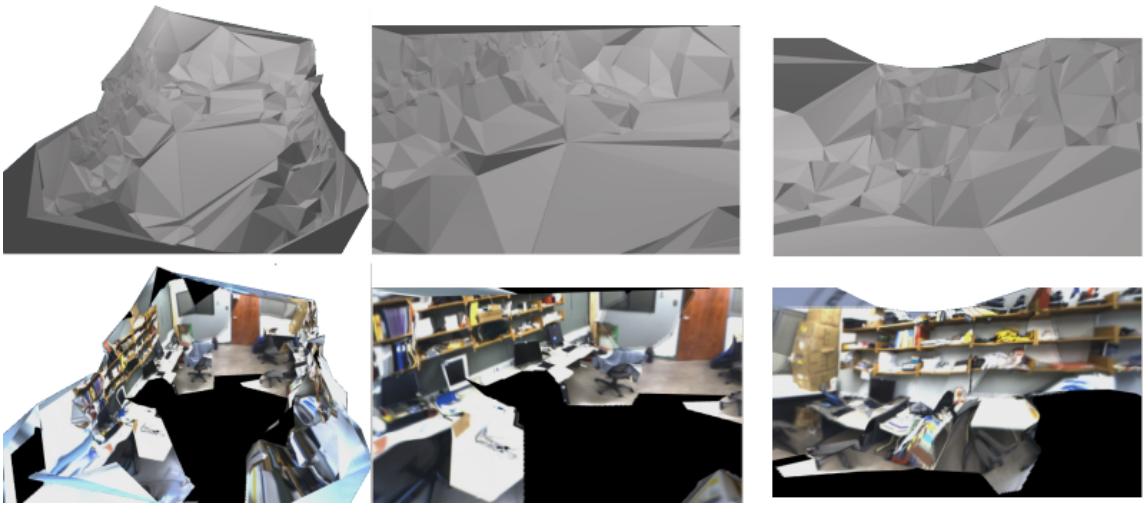


Fig. 6. A model of the Robotics research laboratory constructed with our system moved around the lab. Although the geometries are coarse, this does not impair the user in performing the tasks in Sections III-B and III-C appropriately. Top: Geometry, shaded. Bottom: Texture overlay, from same viewpoints.

Fig. 7. Align task. Left: image of the setup. Right: the camera view illustrating the overlay (white A) which should be aligned to the real A.

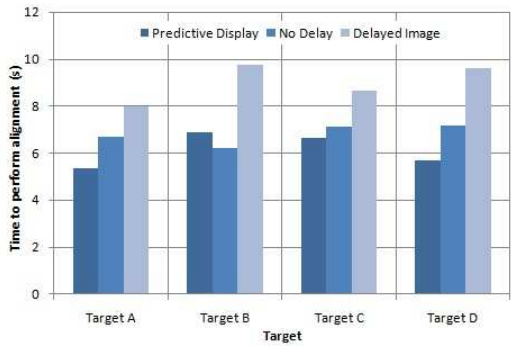


Fig. 8. Mean normalized times to perform the alignment for each target.

system performs well with a real-world model from video. The completion time for the alignment task was improved by the use of the predictive display. It helped the users cope with both the transmission delays and the velocity and acceleration limits of the robot. Fig. 8 shows the mean normalized times to perform each of the alignments. Our statistics are normalized the same as in [13] so that each subject contributes to the results equally.

It is important to note that even though the model is rough and the overlaid texture does not match in detail with the scene's geometry, this did not seem to handicap the user. The users were oblivious to the model not being perfect.

C. Inspection Task

Inspection is useful when evaluating systems' functionality in remote environments (*e.g.*, determining if an electronic board is burnt or evaluating a mechanism's operability after some damage). This last experiment's purpose is to test predictive display in a different task where the user does not perform an action on the environment but assesses the situation from information in the scene.

In this experiment, the setting involved two panels with a 3x3 LED matrix. In most cases, the panel would have one "damaged" unlit LED which is to be identified by the user; in other cases, all the LEDs were lit. These panels were placed in two randomly selected positions out of five possible locations for each trial. Two users were asked to do the inspection in two trials for each visual mode (*i.e.*, six trials per user in total). Again, the mode order was selected at random. The unlit LED in each panel was also random in each trial. In this experiment, the user was allowed to read the panels in any order. Fig. 9 shows how the task looks from the operator's point of view.

The experiment started with the robot in home position. The task was to first identify the position of the two LED panels and then move the camera close enough in front of the panel to identify the unlit LED, if any. From afar, one can locate the lit panels, but it is practically impossible to make an assessment of which LED is unlit.

Fig. 10 shows the mean normalized times for the inspection task. Predictive display improves the ability of the user to cope with both the delay and the dynamics of the robot. It is important to note that for the first panel, the predictive display allows the user to read the panel in less time than when using non-delayed or delayed video streams. Most users looked for both panels from a point of view where the whole scene was visible, yet it took more time to move to a suitable position to read the second panel with predictive display than with non-delayed video. It was observed that because of the single texture used in the predictive display,

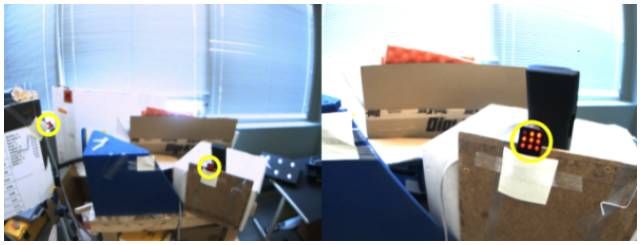


Fig. 9. An operator performing inspection. Target locations are seen from far away [left] but identification of burnt LED requires a close view [right].

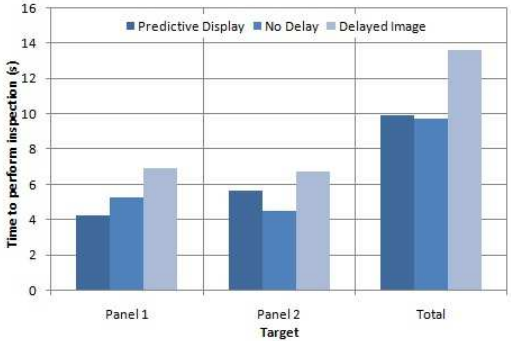


Fig. 10. Mean normalized times to perform the inspection task.

the second target could go out of view when there was no overlaid texture to help with inspecting the second panel.

IV. Conclusions

We have developed a system for predictive display with a real WAM robot that uses online visual tracking and mapping for pose estimation. A PHANTOM Omni haptic device is used to tele-operate the remote WAM robot. The predictive display uses an original incremental real-time free-space carving technique with regularized isosurface extraction to compute a geometric proxy. This model is computed at the remote robot site and transferred back along with images that are used with projective texture mapping in the rendering.

We have obtained reasonably accurate free-space models of unknown environments. Our experiments validate results that were previously only obtained on synthetic data: operators can perform alignment tasks more efficiently with predictive image-based display compared to delayed video. We also found that because the predictive display is not constrained by the robot's kinematic constraints or velocity/acceleration limits, it is possible for operators to achieve better performance with predictive image-based display in delayed situations than using non-delayed video. Furthermore, we have preliminary results suggesting that similar benefits exist in inspection tasks.

A. Extensions / Future Work

Our current hardware setup offers immediate directions for improvement. Our WAM wrist is sent for repair, which is why we have resorted to the 4 DOF configuration. Incorporating the extra DOF should be straightforward. In future work we could use the motor motion signals of the Segway RMP and the WAM to improve robustness of the vision-based tracking and mapping process. Although we use a haptic device for tele-operation, we do not yet provide any haptic feedback. In the future we would like to analyze the benefits and trade-offs of predictive haptic feedback vs. predictive visual feedback.

In terms of the predictive rendering, we would also like to analyze more elaborate alternatives for rendering, some of which might be better suited to communication channels with large delay or low bandwidth (*e.g.*, such as the dynamic texture [12]). An alternative to improving the texture/appearance component would be to improve our approximate geometry, *e.g.*, using a method that takes photoconsistency into account. These alternatives should be straightforward to integrate into our prototype, as the display, model building, and tracking components of our system are modular and swappable.

References

[1] T. Sheridan, "Space teleoperation through time delay: Review and prognosis," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 592–606, 1993.

[2] R. Held, A. Efstathiou, and M. Greene, "Adaptation to Displaced and Delayed Visual Feedback From the Hand," *Journal of Experimental Psychology*, vol. 72, no. 6, pp. 887–891, 1966.

[3] S. Ellis, M. Young, B. Adelstein, and S. Ehrlich, "Discrimination of changes in latency during head movement," in *International Conference on Human-Computer Interaction*, 1999, pp. 1129–1133.

[4] W. Ferrell, "Remote manipulation with transmission delay," *IEEE Transactions on Human Factors in Electronics*, vol. 6, pp. 24–32, 1965.

[5] P. Hokayem and M. Spong, "Bilateral teleoperation: An historical survey," *Automatica*, vol. 42, no. 12, pp. 2035–2057, 2006.

[6] D. Lovi, N. Birkbeck, D. Cobzas, and M. Jägersand, "Incremental Free-Space Carving for Real-Time 3D Reconstruction," in *3DPVT*, 2010.

[7] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," in *IEEE International Symposium on Mixed and Augmented Reality*, 2007, pp. 1–10.

[8] A. Bejczy, W. Kim, and S. Venema, "The Phantom Robot: Predictive Displays for Teleoperation with Time Delay," in *ICRA*, vol. 1, 1990, pp. 546–551.

[9] W. Kim and A. Bejczy, "Demonstration of A High-Fidelity Predictive/Preview Display Technique for Telerobotic Servicing in Space," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 698–708, 1993.

[10] T. Burkert, J. Leupold, and G. Passig, "A Photorealistic Predictive Display," *Presence: Teleoperators & Virtual Environments*, vol. 13, no. 1, pp. 22–43, 2004.

[11] M. Jägersand, "Image based predictive display for tele-manipulation," in *ICRA*, 1999, pp. 550–556.

[12] K. Yerex, D. Cobzas, and M. Jägersand, "Predictive Display Models for Tele-Manipulation from Uncalibrated Camera-capture of Scene Geometry and Appearance," in *ICRA*, vol. 2, 2003, pp. 2812–2817.

[13] A. Rachmielowski, N. Birkbeck, and M. Jägersand, "Performance Evaluation of Monocular Predictive Display," in *ICRA*, 2010, pp. 5309–5314.

[14] D. Cobzas, M. Jägersand, and H. Zhang, "A Panoramic Model for Remote Robot Environment Mapping and Predictive Display," *International Journal of Robotics and Automation*, vol. 20, no. 1, pp. 25–34, 2005.

[15] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.