Real-Time Photorealistic Virtualized Reality Interface for Remote Mobile Robot Control

Alonzo Kelly
Nicholas Chan
Herman Herman
Daniel Huber
Robert Meyers
Pete Rander
Randy Warner
Jason Ziglar
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890, USA
{name}@rec.ri.cmu.edu

Erin Capstick
DCS Corporation.
1330 Braddock Place,
Alexandria, VA, USA, 22314

Abstract

The task of teleoperating a robot over a wireless video link is known to be very difficult. Teleoperation becomes even more difficult when the robot is surrounded by dense obstacles, or speed requirements are high, or video quality is poor, or wireless links are subject to latency. Due to high quality lidar data, and improvements in computing and video compression, virtualized reality has the capacity to dramatically improve teleoperation performance – even in high speed situations that were formerly impossible. In this paper, we demonstrate the conversion of dense geometry and appearance data, generated on-the-move by a mobile robot, into a photorealistic rendering model that gives the user a synthetic exterior line-of-sight view of the robot including the context of its surrounding terrain. This technique converts teleoperation into virtual line-of-sight remote control. The underlying metrically consistent environment model also introduces the capacity to remove latency and enhance video compression. Display quality is sufficiently high that the user experience is similar to a driving video game where the surfaces used are textured with live video.

1 INTRODUCTION

Effective operation of any mobile platform without direct line-of-sight is intrinsically difficult to achieve. Conventional vehicle teleoperation works by transmitting one or more video feeds from the vehicle to a remote operator. Viewing the world through the "soda straw" of a video camera causes a reduction or loss of peripheral vision. Once an object leaves the field of view, the operator must rely on his/her memory and motion perception to estimate its location thereafter. Furthermore, operators have limited ways of judging the relative size or position of the vehicle with respect to environmental elements, although some context is possible if part of the vehicle is visible in the image.

Wireless communication links are also subject to dropouts and high levels of latency. If the transmission link between the vehicle and operator is interrupted, the operator has no visual or positional feedback from the vehicle, and this can be very alarming and dangerous when the vehicle is moving at useful speeds. Even without dropouts, high latencies make steering difficult, since the control decisions must be made using outdated information.

The bandwidth limitations of wireless systems typically cause a large reduction in image quality relative to the fidelity of the underlying video cameras. Such reduced fidelity imagery makes it difficult to identify both obstacles and objects of interest.

When the robot undergoes significant or abrupt attitude changes, the operator response may range from

disorientation, to induced nausea, to dangerous mistakes. The need for high attention levels also deprives operators of the capacity to pay attention to both their and the vehicle's surroundings.

All of these limitations increase the operator's frustration and workload, while reducing driving performance. They also endanger valuable assets while increasing the time required to become a skilled operator.

1.1 Technical Approach

We consider the main contributions of our work to be the production and remote display of real-time realistic 3D models of a moving platform and the use of a continuously predictive display for latency compensation.

Some of the problems of video-based teleoperation can be mitigated by creating, in real-time, a photorealistic 3D model of the environment surrounding the vehicle. The term *virtualized reality* refers to the production of views of a rendering model where the geometry and appearance content is derived from measurements of a real scene. For visualization, such a model makes it possible to render synthetic views of the scene from arbitrary perspectives that may never have been the site of any real sensor.

Such techniques represent an extreme on a spectrum of real data content with augmented or mixed reality somewhere in between and virtual reality at the other extreme. Virtualized reality enables a new capacity to address many of the problems described above by providing a photorealistic, synthetic, line of sight view to the robot based on the content of geometry-augmented real-time video feeds.

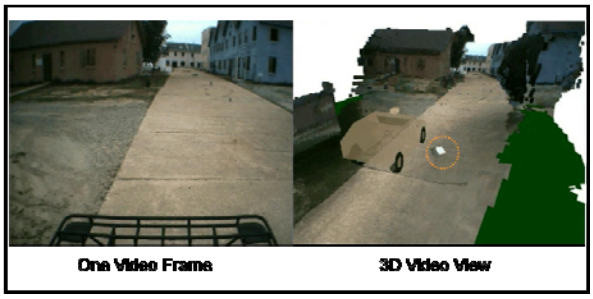

Figure 1: 3D Video View of a Mobile Robot. Left: A video frame produced from a camera on a moving vehicle. Right: The 3D Video view produced from all of the video that has been received in the last few seconds by the vehicle. The operator can look at this model from any angle, at any zoom, while it continues to be updated in real time as the vehicle moves in the scene. The vehicle is synthetic since no sensor has imaged it, but the rest of the content is generated from live video produced by the forward looking sensor mounted on the vehicle's roof.

When virtualized reality processing is performed in real-time, a kind of hybrid *3D Video* (Figure 1) is produced that can be viewed from arbitrary perspectives while exhibiting the photorealism and dynamics of live video. The operator experience is equivalent to following the robot in a virtual helicopter that provides arbitrary viewpoints including an overhead viewpoint and the over-the-shoulder view that is popular in video games.

We use lidar data to sense range and color cameras for appearance. The engineering difficulty of producing and processing such data is considerable, and it is even more difficult if it must be photorealistic and produced in part from data feeds of scanning lidar sensors derived from a continuously moving vehicle in natural terrain. Nonetheless, the rewards of such efforts are also considerable as we hope to show in the sequel.

A clear advantage of our approach is the capacity to produce the viewpoint that an operator would prefer, and even allow it to be changed to adapt to the task at hand. The overhead viewpoint of Figure 2, for example, is ideal for parking while driving in reverse with no rear camera.

Figure 2: Synthetic Overhead Viewpoint. An overhead view can simplify certain operations. Reversing the vehicle into a parking space without the benefit of a rear-facing sensor is shown in this sequence of three images.

While the effects of latency on teleoperation performance are well known, recent work by our colleagues (Ross, Bares, Stager, Jacker, & Perschbacher, 2008) has explored the question of how best to allocate video, display, and communications resources. This work is based on using a fiber optic tether to produce a kind of ground truth teleoperation experience. In one of their results, repeated here as Figure 3, the difficulty involved when humans try to compensate is evident.

Figure 3: Effects of Video Latency on Teleoperation Performance. For even expert drivers, an increase in latency from 360 msec (green) to 1300 msec (red) doubles driving time, halves average speed and produces a system declared "impossible" to drive.

Our approach of committing to virtualize the entire scene in real-time eliminates the difficult problem of mentally distorting a delayed video sequence to represent what it should look like "now". A corrected interior (out-the-window) view is produced by simply placing a synthetic camera at the correct pose. Conversely, a corrected exterior view is produced by rendering a virtual vehicle at the right pose relative to its surroundings.

1.2 Motivation

In addition to better, more realistic views and reduced latency, a large number of other benefits can be realized with our approach to user interfaces. A more complete list of these benefits includes.

- The operator sees the entire hemisphere around the vehicle at all times – permitting display of objects outside the instantaneous camera field of view.
- The display provides a natural mechanism to introduce augmented reality operator aids.
- The viewpoint is stabilized regardless of the attitude (pitch and roll) changes of the real vehicle.
- Viewpoints can be customized and switched for each task.
- Multiple viewpoints can be shown at once, and multiple operators or observers can view the same scene model independently.
- Objects in the environment can be analyzed geometrically, for example, to determine if a path between obstacles is wide enough to pass through.
- Objects can be examined closely by zooming in on them.
- The frame rate of the display can be adjusted independently from the underlying video feed.
- Dropped frames can be easily tolerated by rendering the most recent model.
- Deliberate dropping of frames or many other schemes can be used to compress the data transmission.
- Even with a vehicle moving in the display, latency can be essentially eliminated by rendering the vehicle at its predicted position in the scene.
- A photorealistic map of the area traversed is produced as a byproduct of system operation.

Smallman and St. John argue that improved realism is not necessarily a benefit, because highly realistic displays can actually make it more difficult to sift out the important aspects of a situation – a concept they termed "naïve realism" (Smallman & St. John, 2005). In the case of vehicle teleoperation, realistic visualization is critical for enabling the driver to accurately interpret the scene and to make split-second control decisions correctly. Nevertheless, our virtualized reality approach supports the design principles for avoiding naïve realism. For example, our environment model enables augmentation of scene aspects, such as obstacle locations, which may be easily missed by the operator.

1.3 Related Work

The technique of *view interpolation* (Chen & Williams, 1993) was introduced in computer graphics as a mechanism to efficiently produce synthetic imagery while bypassing many of the computationally expensive geometric aspects of rendering. By exploiting the fact that the depth of each pixel is known perfectly in synthetic imagery, the technique produced synthetic views by linearly interpolating disparity maps and rendering pixels in back to front order based on the reference image depth map.

Various methods have been used for generating virtual viewpoints over the years. View interpolation is one of several approaches to *image-based rendering*. Such techniques achieve remarkable realism through the use of natural imagery to texture surfaces. Image-based rendering techniques allow novel views to be synthesized from images only, but the methods are limited to viewpoints close to or between camera viewpoints (McMillan & Bishop, 1995). Although approximations were originally used for efficiency in view interpolation, the knowledge of the depth of every pixel in a real scene makes it possible to compute an exact mapping to a new image by warping an image according to its depth map.

Accordingly, a purely synthetic view of a real scene can be produced by projecting the pixels of an image to their proper 3D locations and re-projecting them onto a new image plane.

While the earliest approaches to image based rendering used assumed range data or refined approximate range data (Debevec, Taylor, & Malik, 1996), eventually computation was adequate to compute depth maps from the imagery itself. Recent work (Hoiem, Efros, & Hebert, 2005) has even shown that 3D can be extracted from a single image but such techniques are not as accurate as lidar and they do not operate in real-time.

Kanade coined the term *virtualized reality* (Kanade, Rander, & Narayanan, 1997) to emphasize that the image data was natural rather than the synthetic imagery used in virtual reality. Initial virtualized reality work (Rander, 1998) was based on stereo ranging and stationary sensors that surrounded a localized moving scene. It was not possible to digitize the video from the 51 cameras used in real-time, let alone compute the necessary models. However, this work clearly demonstrated the basic mechanism of 3D video – to texture accurate dynamic geometry with dynamic real textures in order to produce a "video" that can be viewed from a synthetic cameras placed anywhere.

More recently real-time image-based rendering has been accomplished for a single discrete object and fixed cameras based on a visual hull method for computing depth (Matusik, Beuhler, Raskar, Gortler, & McMillan, 2000). As in the earlier work, both the geometry and the textures were extracted from the natural imagery, and the result could be rendered from any viewpoint while potentially exposing holes in the sensor coverage due to occlusion.

Computer vision techniques have also been applied to the problem of building models of expansive areas. Some work has focused on individual buildings or terrain models (El-Hakim, Boulanger, Blais, & Beraldin, 1997) (Schneider & Klein, 2008) (Stamos & Allen, 2002) while attempts to build photorealistic models of entire cities have also been ongoing for about a decade (Früh & Zakhor, 2004) (Ho & Jarvis, 2007) (Hu & Neumann, 2003). This community has faced many similar challenges to those that we face, including coping with disparate views, occlusion, and missing parts. Some efforts have fused laser scanner and camera data, whereas others (Mordohai, Frahm, & Akbarzadeh, 2007) (Se & Jasiobedzki, 2008) use stereo and video. In general, the last decade of effort has struggled to achieve full autonomy and real-time performance. All systems still operate off-line using batch data or they make key algorithmic assumptions that limit their use to urban environments.

While elements of computer vision and computer graphics evolved toward virtualized reality, telerobotics was simultaneously developing augmented reality displays for more effective remote control. Numerous techniques for supervisory control and teleoperation of manipulators, and even telepresence, were clearly outlined as early as the mid 1980s (Sheridan, 1986). Our work can be considered an example of model-based teleoperation (Funda, Lindsay, & Paul, 1992), known then as *teleprogramming*. Virtual displays that are either predictive or used for preview have often been used to compensate for both delay and limited data bandwidth when remotely operating manipulators. The models have not been photo-realistic until very recently. A more intuitive and task-centric interface to a manipulator, operating over thousands of miles of separation, was demonstrated in (Lloyd, Beis, Pai, & Lowe, 1997). That effort is an early example of both the use of scene analysis to improve the accuracy of the display and of task-centric interfaces.

The potential of augmented reality environments has been explored in both nuclear servicing (Milgram, Yin, & Grodski, 1997) and space (Kim, 1993) (Kim, 1996) contexts. In these cases, a small amount of virtual information was rendered over natural video. Enhanced accuracy of the state of the models used was achieved using registration of the virtual model to the real remote scene. Augmented reality has also been utilized in the aviation industry, where synthetic vision systems have been used, for example, to reduce occurrences of controlled flight into terrain (Theunissen et al., 2005). Such systems provide an integrated visualization of an airplane's planned path, deviations from the planned path, and the path's relation to the terrain.

Latency compensation in space applications has been accomplished with motion preview and predictive displays. Such displays permit motions to be designed and planned in non real-time before the manipulator is permitted to execute the motion. In some cases, stereo graphics viewed in a stereoscopic display have been used to improve operator depth perception (Milgram, Zhai, Drascic, & Grodski, 1993). While all of the telerobotics work described so far has been applied to stationary manipulators in a stationary scene, the principles are extendable to moving sensors in a dynamic scene, if the image processing is sufficiently efficient. For example, Ricks et al. used a predictive method that they dubbed "quickening" to compensate for latency when teleoperating a mobile indoor robot (Ricks, Nielsen, & Goodrich, 2004).

Model-based interface concepts were also considered early for legged vehicles (Messuri & Klein, 1985) and wheeled Mars rovers (Chatila, Lacroix, Simion, & Herrb,

1995). Given the sensor data needed, the earliest approaches to vehicle teleoperation simply displayed the raw sensor data or showed the robot in a 2D overhead view in the context of its surrounding perceived objects. Applications like space exploration generated a strong impetus to develop more realistic virtual displays as early as 1991 (Hine, Stocker, & Sims, 1994).

One sustained research effort in the use of virtual environments for the control of robot vehicles is the Virtual Environment Vehicle Interface (VEVI) described in (Hine, Hontalas, Fong, Piguet, Nygren, & Kline, 1995). This system was tested terrestrially (Fong, Pangels, & Wettergreen, 1995), and derivatives were ultimately used on the Mars Pathfinder mission. Contemporary developments include more emphasis on sensor fusion (Fong, Thorpe, & Baur, 2001) as well as efforts that display appearance and geometry in a less integrated but more useable way (Ricks, Nielsen, & Goodrich, 2004) (Ferland et al., 2009).

Of course, virtualized reality teleoperation depends on the use of adequate sensing. Military and consumer markets have driven the development of some sensors that are relevant to mobile robots today: guidance systems and TV cameras. However, while laser ranging sensors are now commercially produced for factory robots, systems designed specifically for outdoor mobile robots are either single axis, immature products, or of inadequate performance for our purposes. For these reasons, our work continues a long tradition in robotics (Lewis & Johnston, 1977) (Ryde & Hu, 2008) of custom sensor development for lack of any alternative. Such activity continues in robotics labs around the world up to the present time (Möller, Kraft, Frey, Albrech, & Lange, 2005).

Mobile robots already need to build useable models of their surroundings while operating in natural terrain at speed. For this reason, our approach is able to borrow many ideas from contemporary robot autonomy algorithms. Recent outdoor autonomous vehicles compile volumetric representations (Lacase, Murphy, & DelGiorno, 2002) before searching for the supporting surface (Wellington & Stentz, 2004). Methods for predicting motion under a terrain-following constraint are fundamental to obstacle avoidance and they have been used since the first outdoor mobile robots (Daily, et al., 1988).

1.4 Discriminators

Given the long history of research efforts that precede the work presented here, it is worthwhile to mention how the present paper is distinct from this history and ongoing work. Our work presented in this paper uses similar techniques to those that have been used by others for remote robot control. However our work is distinct in its capacity to produce photorealistic displays at a rate and quality level that is sufficient to drive a vehicle at high speeds in arbitrary traversable terrain.

The VEVI system is a clear landmark in related work and it is closest to the work we present here. Our work is distinct from VEVI in that VEVI did not use real video to texture surfaces, and hence did not use virtualized reality. VEVI did render false color terrain maps produced by on-board lidar sensing for a slow moving legged vehicle and this achievement was unprecedented using the technology of that period. Viz, a more recent visualization environment developed by the same group, incorporated imagery from stereo cameras to provide texture-mapped surfaces (Edwards, et al., 2005). VEVI used a classical form of latency compensation based on vehicle autonomy and supervisory control interfaces, but it did not perform the kind of high fidelity continuous motion prediction in virtualized reality that we will present here. We also achieve results in data compression and unprecedented vehicle speeds that derive respectively from the commitment to virtualize the entire scene and the use of custom photogeometric sensing as described below.

Two recent works in mobile manipulator control are similar to our work. In (Johnson, Alberts, & Edwards, 2008) a photorealistic system which is focused on manipulation is presented. Here, the triangular and quad mesh models are produced in non real-time and there is no discussion of whether the model is acquired or rendered while the vehicle is moving. There is also no discussion of the pose solution quality or the nature of the terrain. Unlike this work, we also generate models far from the vehicle and integrate them over excursions on the kilometer scale. In (Buss, Peer, Schaub, Stefanov, & Unterhinninghofen, 2008) a more recent virtualized reality teleoperation system for two mobile dual-manipulator systems is presented. Data from a PMD lidar and a color camera is fused to produce photorealistic models to demonstrate the remote manipulation task of repairing a pipe.

The technical approach of (Mordohai, Frahm, & Akbarzadeh, 2007) is similar to ours in that a high performance INS-GPS system is used and GPUs are used to accelerate processing to the point of achieving real-time performance for multiple high resolution cameras. However, that effort is focused on building maps offline, even if the algorithm is real-time, and it uses stereo ranging in an urban environment. Our challenges of processing lidar data on a vehicle moving rapidly in natural terrain are equally difficult, but different than stereo mapping in cities.

Several research efforts have focused on mobile robot control in indoor environments. Nielsen et al. projected the image from a monocular camera into the visualized

environment along with 3D positions of obstacles detected by a horizontally mounted planar lidar (Nielsen, Goodrich, & Ricks, 2007). Ferland et al. developed a similar interface that replaced the projected monocular image with a 3D surface derived from a stereo vision system (Ferland et al., 2009). Neither of these approaches achieves a level of situational awareness that would allow high speed vehicle teleoperation, and they are best suited to relatively benign indoor environments.

One recent effort (Se & Jasiobedzki, 2008) is vehicle based. A man-packable robot is fitted with stereo cameras which implement a SIFT-augmented structure from motion solution. This system was about one fifth real-time based on a single reduced resolution camera feed and it was focused on the production of a map rather than its real-time use in teleoperation.

Our focus on presenting displays to a human operator makes photorealism and timeliness far more important than geometric accuracy. We also must operate in terrain which is highly self occluding, at times, while attempting to provide the operator with the capacity to view the scene from perspectives very different from those of the original sensors.

While autonomy systems continue to advance, we address here the problem of driving vehicles remotely at high speeds when only a human can presently be trusted to do the driving. Supervisory control interfaces are somewhat feasible in such applications but they can only reduce concentration levels somewhat.

The problem of driving a vehicle continuously is inherently real time so off-line motion preview to compensate for latency is not possible. We do generate a kind of predictive display in order to compensate for latency. In contrast to historical manipulation systems however, ours is continuous, photorealistic, and based on relatively high fidelity models of a ground vehicle. Our predictive display is also completely virtualized in the sense that the operator never sees the raw video upon which a classical predictive wireframe display like (Bejczy, Kim, & Venema, 1990) might be drawn. In our case, the operator sees the vehicle essentially where it is now regardless of the latency in the state feedback.

We address arbitrary terrain where photorealism is necessary to assess its "traversability" (the capacity to be driven over without harming, impeding, or entrapping the vehicle). We use custom sensors in order to generate adequate video and range data.

Our robot is not confined to a small space surrounded by sensors like much historical work but rather we generate displays from sensors mounted on the platform, and the robot moves entirely beyond the effective range of those sensors every few seconds. Also, the robot may be kilometers away from the operator. The speed of the robot generates a requirement to precisely track the motion of the platform in order to integrate the data streams generated as the vehicle bounces around on the terrain, and this problem is acute for the lidar range sensors we use in order to achieve photorealism.

1.5 Organization

This paper is organized as follows. Section 2 describes our custom photogeometric sensor concept that enables the production of 3D Video. Section 3 describes the perception sensors, the navigation system, the semi-autonomous vehicle, the operator interface, and the overall system architecture. Section 4 describes the key algorithms used for modeling and visualization. Section 5 discusses how such algorithms are augmented with latency compensation, vehicle control, and telemetry compression mechanisms to produce a complete teleoperation system. Section 6 presents results from one week long user study designed to quantify operator remote control performance improvements. Section 7 presents a brief conclusion.

2 Photogeometric Imagery

The term *appearance* will be used to refer to sensing modalities that are sensitive to the intensity of incident radiation including visible color, visible intensity, and infrared modalities. Conversely, *geometry* will be used to refer to modalities that register any of depth, range, shape, disparity, parallax, etc. The term *photogeometric* (PG) sensor will refer to a sensing device that produces both kinds of data in a deeply integrated manner. For our purpose in this paper, the data is deeply integrated if the spatial correspondences of the data are known. Ideally, as shown in Figure 4, the resolutions are matched as well so that a one-to-one mapping exists between geometry and appearance pixels.

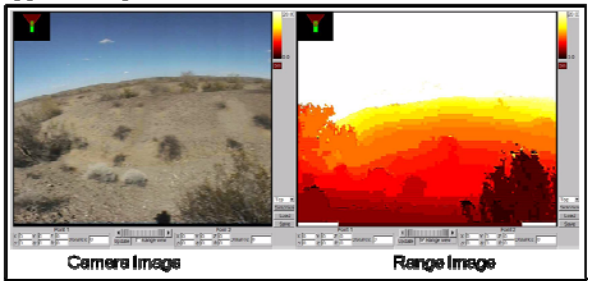

Figure 4: Photogeometric Data Set. Every color pixel in the left image has an associated range pixel in the right image. Adequate sensors that produce such data do not exist on the market today but they can be constructed by integrating more basic components.

At some point in the future, flash lidar devices may be available that share apertures with color cameras in order to produce photogeometric data in hardware. Until that day comes, we find the value of PG data to be worth expending effort to produce it in whatever manner we can today.

Our sensor implementation approach centers on the goal of producing an integrated data set of appearance and geometry data from two distinct sensors. The data may be organized arbitrarily but our two most common formats are camera-derived color data augmented with range, "*rangified color"* (RC), and lidar-derived range data augmented with color, which we call *colorized range* (CR) data.

Computational stereo vision is a natural RC modality because range is produced for every pixel in the reference appearance image. However, its utility in applications can be limited due to the relatively poor quality of the range data. Stereo has quadratic uncertainty variation with range, limited practical range, and a trade-off between accuracy and field of view that prevents accurate wide angle stereo from a single pair of cameras.

Flash lidar sensors also continue to advance (Anderson, 2005) but none yet meet our requirements for operation in outdoor environments. Conversely scanning lidar devices have been our preferred geometric sensing modality for two decades. We will therefore discuss PG sensing where the range data is provided by a scanning lidar.

In general, every appearance modality can potentially be paired with every geometry modality. Ideally, each sensor of a pair would image the same region of the scene as the other, at the same resolution and frame rate, and from the same position. In practice, numerous technical issues arise due to the different attributes of the two sensors including:

- **Projective Geometry**. Lidar is often spherical polar, whereas cameras (and flash lidars) provide a perspective projection.
- **Resolution**. Scanning lidar typically produces 1% of the angular resolution (solid angle) of a camera so there can be up to 100 camera pixels for each lidar measurement.
- **Field of View**. Standard camera lenses, spherical mirrors, and lidar scanning mechanisms rarely provide the same field of view.
- **Baseline**. Displacement of one sensor center of projection or emission relative to another leads to parts of one view missing from the other – even if all other parameters match.
- **Frame Capture and Beam Scanning**. In cases where data is gathered on the move, each point of lidar data is captured from a different sensor position whereas all pixels in a camera frame come from a single position.

2.1 Establishing Pixel Correspondences

A basic property of cameras is their projective geometry that projects a 3D scene onto a 2D photosensitive sensor array. While the azimuth and elevation coordinates in the image are related to the equivalent directions in the scene, information about the depth of objects is lost when a camera image is formed.

Hence, the most valuable attribute of PG imagery is its recovery of the depth dimension that is lost when a real scene is imaged with a camera. This information is recovered by:

- establishing an association of lidar range points with camera pixels
- geometric transformations to convert lidar data to camera coordinates

For RC data, the color data is augmented with depth so that the result is an augmented image. For CR data the range data is colorized, and the result is an augmented range image or point cloud. In either case, the mechanism to establish correspondences is the same as discussed below. For now, suppose that both sensors are stationary with respect to the scene, and let us define a lidar "image" to mean the data produced by one sweep over the scene of the lidar scanning mechanism.

Consider Figure 5 that expresses the essence of the problem when both sensors are viewed from overhead. While it is not clear how to directly map color pixels onto a lidar data set, the reverse operation is conceptually straightforward if we ignore issues of angular resolution matching between the two sensors. Hence both RC and CR datasets rely on a common procedure to establish correspondences. Let the letter L designate a coordinate frame attached to the lidar center of emission, and let the letter C designate one at the camera center of projection. The homogeneous transform matrix that converts coordinates of a point from frame L to frame C is denoted T_L^C. Let the letter I designate row and column coordinates in the camera image plane. The projective transformation matrix that provides the image coordinates of a 3D point will be designated P_C^I. The homogeneous dimension will be omitted from vectors unless the matrices are written out. Under this notation, the camera image coordinates $\underline{r}^I = [x \quad y]^T$ of the point imaged by a lidar point $r^L = [x \quad y \quad z]^T$ are:

$$\underline{r}^I = P_C^I T_L^C \underline{r}^L \tag{1}$$

If the scene has sufficient 3D (non-planar) structure, the spatial separation of the sensors introduces characteristic problems of triangulation:

- **Missing parts**. Even with perfect field of view overlap, surfaces oriented perpendicularly (and hence invisibly) to the viewing direction of one camera may be visible to the other. Regions near depth discontinuities have similar issues.
- **Depth ambiguity**. It is possible for the lidar to have ranged to a point on a background object that is behind a foreground object that was imaged by the camera.

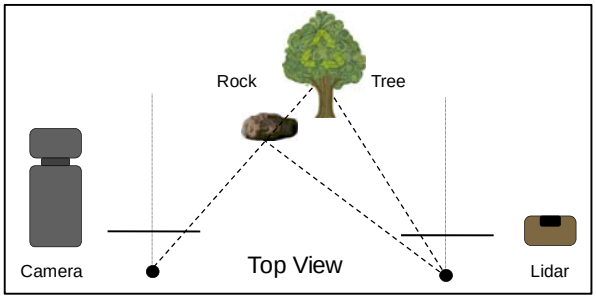

Figure 5: Multi Sensor Geometry and Depth Ambiguity. The camera measures the angle to objects whereas the lidar measures angle and range. It is straightforward to project a range point onto the image plane. Due to the baseline separating the sensors, a lidar may image more than one object along the line of sight of a camera pixel.

While the first problem has no solution, the second can be solved by forming a depth buffer of all of the lidar data as viewed from the perspective of the camera image. All lidar data can be projected into bins that are sorted by depth or the processing may simply retain only the smallest range value in each bin. In either case, when two or more lidar pixels fall on the line through a given camera pixel, only the closest lidar point should be associated with the color pixel. All others are occluded and invisible to the camera, so their color is unknown. Limited lidar resolution relative to cameras adds an extra level of complexity.

While these triangulation issues cannot be eliminated entirely, they can be mitigated significantly by placing the two sensors very close together relative to the depths being imaged.

2.2 Forming Photogeometric Datasets

Whereas the last section considered only the image formation geometry, this section considers angular resolution issues. Given the correspondences between range and appearance data, either CR or RC data may be formed. The production of CR data using lidar is easiest to illustrate. In this case, the sensor intrinsic data format is a temporally ordered set of 3D points expressed in Cartesian or polar coordinates relative to the sensor center of emission. Each lidar point is simply augmented by the color of its associated camera pixel, if any. The color information might be the color of the closest camera pixel, the average over a region around it, or a block of pixels forming a small texture map.

In the case of RC data, the goal is to produce range data for every color pixel in a color image. Typical camera angular resolutions are 1 millirad whereas lidar is typically 10 millirad. Hence, once the lidar correspondences are computed, only 1% of the camera pixels can be expected to have associated lidar points. In other words, there will inevitably be holes in the coverage of the image by the range data. Small holes will be due to the reduced angular resolution of the lidar and larger ones due to occlusion or non-overlapping fields of view.

When dense range data is desired, range interpolation can be justified on the basis that the lidar is really providing the average range of the region of the scene that is spanned by a large number of camera pixels. The range data can be interpolated using the dilation operation of computer vision to fill small holes. The dilation radius can be related to the expected angular lidar footprint in the camera image. When both sensors are close together, the effect of surface orientation on pixel footprint prediction is minimal. Note, however, that it is also important to avoid range interpolation across depth discontinuities.

2.3 Precision Timekeeping and Pose Tags for Moving Sensors

PG sensing was originally motivated by its capacity to disambiguate natural obstacles and non-obstacles of the same shape (such as a rock and a bush (Dima, Vandapel, & Hebert, 2004)) by examining their color signatures (Figure 6). Once we had such data available for use in autonomy, we began to produce specialized point cloud displays, and we quickly recognized the potential of the PG data for human interfaces (Anderson, Howard, Apfelbaum, Herman, & Kelly, 2008).

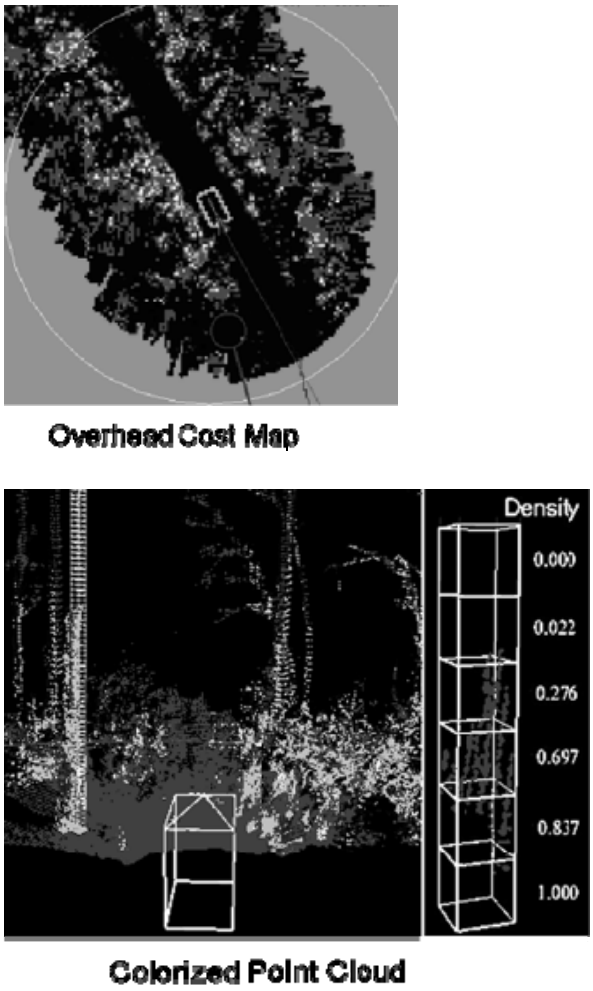

Figure 6: Displays on Autonomy Programs. The display of "traversability" / cost or elevation from an overhead display (top) is traditional in mobile robotics. In recent years, colorized point clouds have also been used. The evolution of the bottom figure toward photorealism was a natural extension of ongoing efforts.

A key difficulty in producing geometrically consistent datasets is the matter of accounting for the motion of the lidar during the sweep of its scanning mechanism over the terrain. For 30 Hz video, individual lidar measurements may occur up to 1/60 of a second before or after the instant of acquisition of the closest camera image.

Furthermore, each lidar measurement in a time sequence has a different time offset from its closest camera image. This variation combined with the vehicle motion between camera frames means the center of emission of the lidar is different for each lidar measurement, and it is moving relative to the center of projection of the camera for the corresponding camera image.

The basic procedure used to establish correspondence is to compute the pose relating the sensors to each other based on the known vehicle poses at the times that the camera image and the lidar measurements were respectively taken.

Let W denote a world coordinate frame (fixed in the scene). Let the frame corresponding to the lidar pose at the instant of a camera image acquisition be denoted $L1$. Let the frame corresponding to the particular instant that a lidar measurement is acquired be denoted $L2$. In this case, the homogeneous transform matrix T_L^C from equation (1) is computed as follows:

$$T_L^C = T_{L2}^C = T_{L1}^C[T_{L2}^{L1}] = T_{L1}^C[(T_{L1}^W)^{-1}(T_{L2}^W)] \qquad (2)$$

Hence, the pose of the lidar in the world frame must be known at both instants in time. This depends on the pose of the vehicle in the world and the pose of the lidar on the vehicle. Typically, the vehicle navigation system is a composite inertial and satellite navigation system (INS-GPS) which reports its pose, rather than that of the vehicle, so the pose of the navigation system relative to the vehicle must be introduced into the calculations as well.

The engineering difficulty of a precision implementation of these ideas is substantial. We need to track the motion of a lidar very precisely in all 6 degrees of freedom of rigid body motion. Doing that requires excellent short term relative accuracy of pose, particularly attitude and heading, as well as precise measurements of the timing of measurement events.

It is not unusual for the sensors and navigation system to be connected over data links to sensors or even other computers that introduce significant delays in the communication pathway before all of the data exists at one computational node. Therefore, our solution is to precisely synchronize the clocks on all computers. Although any time standard will do, we use the GPS time generated at the navigation system node. Sensor electronics are synchronized to this in order to produce time tags for data as close as possible to the instant the data was acquired.

When the data is finally assembled in one place, it is FIFO buffered to maintain a short time history. The time tags are used to assess the mutual proximity of data acquisition events. Vehicle pose data is not intrinsically available at 10 KHz so we up-sample to this rate using linear interpolation in order to provide the best estimate of vehicle pose for every lidar measurement.

3 Hardware and Architecture

Virtualized reality constructs a computer graphics model of a real scene. The set of geometrically consistent graphics primitives to be displayed will be referred to as the *model*. For teleoperation, a key design decision is the location of the model building process. If it is performed on the vehicle processor, then model updates can be communicated to the remote operator control station and communications bandwidth requirements can presumably be reduced. Reduction is possible because it takes less bandwidth to transmit a fused model that lacks the redundant content of video.

If modeling is performed on the remote operator control station, raw sensor data must be communicated, and bandwidth requirements are higher. Despite this bandwidth cost, we chose the second option (Figure 7) in our implementation. This was done, in part, because the latency compensation process, discussed later, would be more straightforward because operator commands to the robot can be observed (at the operator control station) without significant latency.

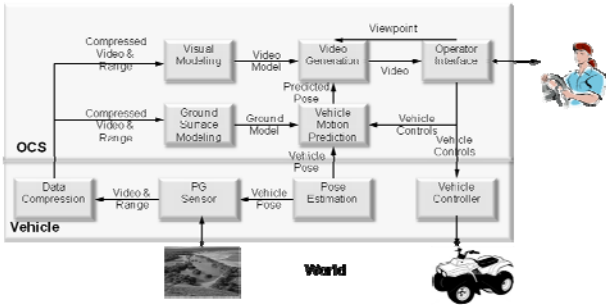

Figure 7: System Architecture. The system includes an operator control station (OCS) and a remote-control retrofit of a standard all terrain vehicle.

The basic hardware setup at the system level involves an operator control station (OCS) that communicates over wireless to a remotely located mobile robot equipped with photogeometric sensing.

3.1 Sensor Configuration

We have been continuously refining our photogeometric sensor concept for many years. Two recent sensor designs are shown in Figure 8. For scanning lidars, we typically purchase an off the shelf scanning lidar that scans in one degree of freedom (called the *fast axis*), and then we actuate the housing in a second orthogonal degree of freedom (called the *slow axis*) in order to produce a scanning pattern that spans a large angle in both azimuth and elevation. For flash lidars or stereo ranging systems, the interfaces to these devices are equal or similar to those of cameras, so the process is more straightforward.

For this system we used scanning lidars for geometry measurement and offset cameras for appearance measurement. We considered using flash ladars, but available flash ladar sensors had insufficient sensing range, and inadequate performance in bright sunlight. We considered using stereo, but ranging performance with real time systems tends to be inadequate in the unstructured, sometimes low-texture regions common to our operating environments. We considered techniques for axially aligning the scanning ladar and the cameras, but chose the side-by-side sensor placement for greater simplicity.

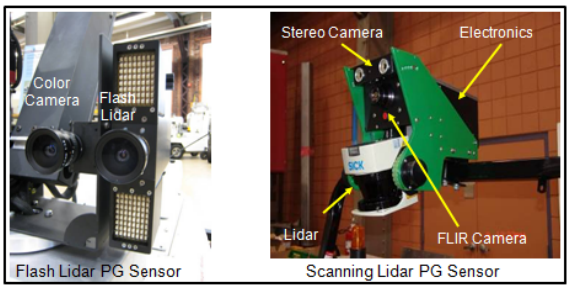

Figure 8: Two Custom Photogeometric Sensors. The device on the right fuses data from a commercial scanning lidar by SICK, stereo cameras, and a forward looking infrared (FLIR) camera. The device on the left fuses a PMD-Tec flash lidar with a color camera. The interface to the composite device is a combination of fast Ethernet (used for high bandwidth data) and CAN Bus (used for low latency control). In the work discussed in this paper, stereo ranging was not performed, but the equipment is used for other purposes where stereo is performed.

The lidar pointing control system provides precisely timed feedback on the angle of rotation. This data stream is merged with the range and angle data coming from the lidar to form a 2D scanning lidar data stream. This stream is then optionally merged with any camera data and transmitted to the host computer system. In autonomy systems, it is often useful to merge the data at the level of individual imagery. However, for visualization, we instead merge the data later, at the level of an integrated geometric model of the scene.

3.2 Vehicle

Our latest vehicle test bed is a custom retrofitted LandTamer® amphibious remote access vehicle, shown in Figure 9. We chose this vehicle for its terrainability, ease of transport, and (deliberately) for the difficulty of modeling its skid steering.

Figure 9: Robot Vehicle. A LandTamer® vehicle was retrofitted for remote control. Three custom colorized range (CR) sensors with a total field of view of 160° are mounted high on the vehicle looking forward. The lidars are manufactured by SICK providing range points at 13 KHz separated by ½ degree of angle over 180°of field of view. The cameras are the Firefly® by Pt. Grey Research Inc., and they provide color imagery at 720 X 500 resolution over a 60 degree field of view.

A custom field programmable gate array (FPGA) board is used to implement the servos that control the nodding motion of the lidars. It is also used to integrate the data into time-tagged colorized range data sets, and to provide the results over an Ethernet link to the main vehicle computer.

Calibration is performed to determine the camera intrinsic parameters and the relative pose between the camera and the laser scanner. This calibration enables 3D points from the laser scanner to be projected into the image to determine the corresponding image pixel. Additional calibration is conducted using a white reference target to correct for vignetting and color differences between multiple cameras. The pose of the composite sensor housing with respect to the vehicle frame is also estimated to allow sensor data to be transformed into world coordinates while the vehicle is moving.

The drive by wire system closes 6 wheel velocity loops based on dual redundant wheel encoders (two on each side) that indicate rotary position of each wheel. The vehicle is driven in differential steer mode – so that all three wheels on one side move at the same velocity. The vehicle hydrostatic drives provide excellent slow speed controllability. Another custom FPGA board interfaces over CAN bus to modulate the positions of valves feeding the hydrostatic drives in place of the joysticks that have the same function on a manually driven vehicle.

3.3 State Estimation and Data Acquisition

A Novatel SPAN INS-GPS system is used for pose estimation. Sensor fusion is accomplished with the vendor's Kalman filter. The system is augmented by a portable real-time kinematic (RTK) differential base station. Under favorable satellite and base station viewing conditions, 2 cm accuracies are achievable over kilometers of excursion.

Of course, an INS-GPS system is an expensive pose estimation solution but we had one available. Also, the short term relative accuracy requirements of the attitude and heading solution are severe since they determine the fidelity of reconstruction of a surface sampled over a (lidar) lever arm many tens of meters long. Sufficiently accurate orientation may only be achievable with a gyro-based solution. GPS is a useful addition as well but it not critical to accurate reconstruction of surfaces. In lieu of GPS, any mixture of odometric dead reckoning and visual guidance, including visual SLAM (simultaneous localization and mapping) may be a viable replacement for GPS for localized reconstruction purposes. Of course, GPS also provides a capacity to accurately acquire geo-referenced waypoints which can be important for other applications.

A small computing cage houses the sensor control and data acquisition FPGA board and two Intel® Core™ Duo processors. These processors concentrate the data from all sensors and send it to the OCS (described below) over 802.11g wireless. They also receive the OCS commands over the same wireless link and pass them to the vehicle controller.

3.4 Operator Control Station (OCS)

The OCS, shown in Figure 10, incorporates a steering wheel, throttle, and brake pedals (Logitech MOMO), as well as a large LCD monitor. Buttons on the steering wheel are used to select various views, and to control convenience features like driving direction (forward or reverse), and velocity cruise control. The OCS processor is an off-the-shelf personal computer (Intel Q6600 quad-core 2.4 GHz CPU, GeForce 8800 Ultra video, and 4 GB memory). It is capable of both communicating with the robot and rendering the data on the display.

Figure 10: OCS. The Operator Control Station includes a steering wheel equipped with selection buttons. It also has foot pedals, and a large LCD display. The display provides selectable views including the raw video, over-the-shoulder, and bird's-eye (direct overhead).

The interface was designed to aid an operator in the performance of outdoor driving tasks. It emphasizes the use of multiple selectable views for driving slowly both forward and backward in proximity to hazards. The interface is also suitable for high speed driving in terrain where high speeds are feasible and safe. While it would be possible to add aids for reconnaissance or search tasks, we have not done so. An example aid for search would be a visual display of regions that have been visited already. Likewise, the interface includes no facilities to support manipulators or other tools, although the potential to produce them (for example zoom in on the end effector and object) is clear.

4 Modeling and Visualization Algorithms

In rough terms, the process of constructing photorealistic models is one of fitting surfaces to the lidar (geometry) data and projecting the camera (appearance) data onto those surfaces. In order to achieve photorealism, we aspire to produce geometry for every camera pixel (*rangefied color*). Once again, the difficulty of implementation is substantial. This section summarizes our approach and it is covered in more detail and precision in (Huber, Herman, Kelly, Rander, & Warner, 2009).

Numerous effects give rise to situations where the color of a scene point is known, though its range is not. For many reasons, lidar data produced on a ground vehicle ceases to be reliable beyond a range on the order of 30 meters. Let the region beyond this range be known as the *far field*, and let that region inside this range be called the *near field*. Even in the near field, the reduced angular resolution of lidar relative to cameras implies that the vast majority of near field color pixels in a camera image will not have a lidar pixel that corresponds directly.

A second important issue is range shadows. It is necessary in general to depth buffer the range data from the camera perspective in order to ascertain which ranged points are occluded by others and therefore have unknown color. When the viewpoint differs significantly from that of the lidar sensor, substantial missing parts in the model become possible.

For our purposes, the required precision of geometry depends on the offset of the viewpoint from the original sensor viewpoint. When the offset is small, substantially incorrect geometry will still look highly realistic. When the offset is large, differences in parallax of scene points from their correct parallax will result in distortion that is noticeable to the operator.

In general, four classes of points can be distinguished. The system uses several strategies described below to process them.

- **Surface and texture known.** This is the easiest case where the texture is projected onto the surface.
- **Only texture known.** In this case, the geometry has to be assumed or the data rejected. Two cases of practical interest are under-sampled smooth surfaces, and regions beyond the lidar maximum range.
- **Only geometry known.** Enough cameras can be used to ensure that this case does not occur – with two exceptions. First, the vehicle does not normally produce a lidar image of itself, but its geometry can be measured or coded offline. Second, regions occluded by other surfaces can be drawn in an unusual color to identify them to the operator, and both sensor separations in space and image separations in time can be minimized to the degree possible to mitigate this effect.
- **Nothing known.** Once the possibility exists to place a viewpoint anywhere, scenes with complex geometry will often show holes in the model that correspond to regions that no sensor was able to see for occlusion reasons. This is the cost of arbitrary viewpoints applied to data imaged from a specific viewpoint. While fortunate earlier views of the occluded area can occur, there is no way in general to generate the missing data. However, the advantages of arbitrary viewpoints can outweigh this imperfection.

Of course, regions of the scene may become unknown over the passage of time when the scene is dynamically changing. In such cases, omnidirectional lidars and cameras may be used to continuously update the view in all directions. Such sensing will mitigate this issue within the range of the lidars, and even beyond it if billboards are used as described below. However, difficulties remain. If a part of the display that has changed is not updated it could lead to a false impression that a moving object (pedestrian, car, animal) has not moved, or has cloned itself one or more times. This could lead to a disastrous decision on the part of the operator. Dynamic scenes are beyond our scope in this paper and they constitute an important research area in their own right.

4.1 Vehicle Modeling and Visualization

The rendering of the vehicle is the simplest case. When a viewpoint is selected in which all or part of the vehicle itself would appear, a virtual model of the vehicle is placed at the correct location and rendered. Determining the correct location can take some effort as outlined later.

While it would be possible to produce a highly realistic model of the vehicle, we have elected to render a less realistic one. The virtual look of the vehicle reminds the operator that this is the one object in the display that is not continuously imaged in real-time.

4.2 Near Field Ground Surface Modeling

The application to ground vehicles justifies the assumption that the environment around the vehicle includes a ground surface and optional objects that may lie on it. In many environments, lidar data is sufficiently dense, out to 20 to 30 meters, to sample the terrain surface adequately for its reproduction. For this reason, the implementation segments all lidar points into those that lie on the ground and those that lie above it.

In forested environments, situations like overhanging branches invalidate the assumption that surface height is a single-valued function of horizontal position. Therefore, all lidar data is initially accumulated in a 3D voxelized, gravity-aligned data structure, called the point cube, before it is segmented. Each voxel counts the number of lidar beams that have terminated in the voxel (called hits).

There are sophisticated ways to solve the "chicken-and-egg" problem of computing ground surface height and classifying range points (Wellington, Courville, & Stentz, 2006) but our application demands a more computationally tractable approach. After each full sweep of the lidar beam, the point cube is analyzed to determine the lowest cell in each vertical column with enough hits to determine a ground surface. The average height of these hits is used to define the ground height at the position of the voxel in a new data structure called the *terrain map*. This structure is a horizontal 2D array arranged into *cells* (20 cm on a side) that store the ground height at the center of each cell. The terrain map accumulates all data from multiple lidar scans. Its spatial extent, like the point cube, is often limited to some adjustable region around the present vehicle position defined in terms of 3D space, distance, or time. However, we have, at times, accumulated kilometers of terrain in the terrain map when the purpose of the experiment was producing the map itself.

Figure 11 (inset) shows a wireframe rendering of the ground surface. Colorized range points determined to be above the ground surface are also drawn without modification. The model of the vehicle is inserted as well.

4.3 Near Field Ground Surface Visualization

Each cell in the terrain map is converted to two untextured triangles that must then be textured from the camera imagery. While equation (1) permits the mapping from lidar points to image pixels, the situation for multiple sensors on a moving vehicle is far more complex than that depicted in Figure 5. The baseline separation between camera and lidar can unfortunately be enlarged significantly due to asynchrony of the camera and lidar during periods of vehicle motion. Also, camera imagery may overlap due to multiple overlapping fields of view or multiple frames captured over time.

Unless depth buffering is performed, the same textures will be painted onto foreground objects as well as those background objects that are occluded by them. This would not be a problem if the terrain map was the only surface in the scene, but there are others above it. Therefore, rather than use equation (1) we initially used shadow mapping (Williams, 1978) to resolve this issue. We later settled on the use of projective texture mapping (Segal, Korobkin, van Widenfelt, Foran, & Haeberli, 1992) implemented in the OCS graphics processing unit (GPU). This approach textures a scene as if the texture map were projected onto the scene by a classical slide projector. The system maintains a list of the most recent images from all cameras. Each image is used to apply texture to the geometry in the scene in temporal order so that cells that fall outside the field of view of more recent images will retain the last texture painted onto them.

Figure 11 shows the textured rendering of the entire near field environment. The ground surface shown in wireframe (inset) is shown textured with the video. Colorized range points determined to be above the ground surface are also drawn without modification. The model of the vehicle is inserted as well. Though not performed here, a lighting model could be used to generate a vehicle

shadow on the terrain, a technique well known to better ground the vehicle to the terrain.

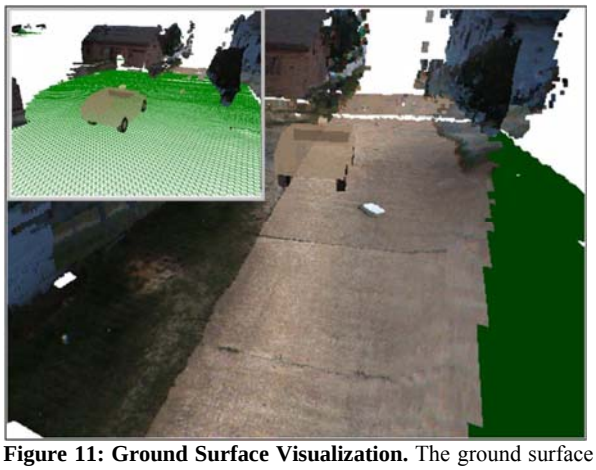

Figure 11: Ground Surface Visualization. The ground surface is estimated using an elevation map, triangulated (inset), and texture mapped. The texture extends behind the vehicle, outside the current sensor field of view, giving the operator historical context.

4.4 Far Field Modeling and Visualization

Often, the far field region of space corresponds to the higher parts of the images, and it extends to the horizon. For such data, our lidars are unable to generate a range measurement, so we erect a temporary surface (a *billboard*) that is normal to each camera's optical axis. The camera data is then projectively textured onto the surface.

The billboards move with the vehicle. Provided the viewpoint is not significantly different from the camera, the parallax error is tolerable, and operators overwhelmingly prefer their use. Figure 12 shows a view of a synthetic vehicle in spatial relationship with the billboards.

Figure 12: Billboards Used to Display Far Field Video. This view shows the geometry of the three billboards and how video frames are projected onto them. The technique of using billboards for complex scenes has been used for many years (Rohlf & Helman, 1994).

4.5 Near Field Non-Ground Modeling and Visualization

Non-ground points in the near field are the most problematic. Any lidar points in a column of voxels that are higher than a threshold from the determined ground height are deemed to be non-ground and are processed separately. When the range data is sufficiently dense, it is possible to interpolate a surface. We have not yet pursued this option because data is rarely sufficiently dense in our case. We presently use two streamlined techniques to render such points.

Often the scene contains relatively high spatial frequencies, and it is severely under-sampled. An example would be a single lidar point landing on a tree branch that is straddled on both sides by "no range" points generated when the beam penetrated the foliage to the sky beyond. So far, our most effective technique has been to render these points as small square surfaces whose size is related to the size of the lidar footprint at the measured range. This process is closely related to the splatting technique (Westover, 1990) for volume rendering. Figure 13 shows an example showing a high-polygon-count tree rendered at high resolution (left) and the point-based rendering approach from a simulated ladar scan (right). While the polygon tree is a superior model, deriving such a model from the limited colorized range data is an extremely challenging problem, especially for a real-time system.

If a voxel contains several spread out hits, this is evidence that a substantial object occupies its volume. In this case, it is acceptable to temporarily assume that the visible faces of the voxel are real surfaces, and then render the camera data on these hallucinated surfaces. We call volumes enclosed by these hallucinated surfaces *legos* after the building blocks toy. The name derives from the block artifacts produced when viewed from sufficient offset from the original sensor location.

Despite these artifacts, legos can be very effective given their capacity to display full resolution video on a surface at the correct range. As long as the lego surfaces remain in the field of view, the texture is updated at high rates, and the viewpoint offset remains low and the display remains highly realistic. Figure 14 shows an example in which the vehicle drove into a cul-de-sac with 3 solid walls around the vehicle.

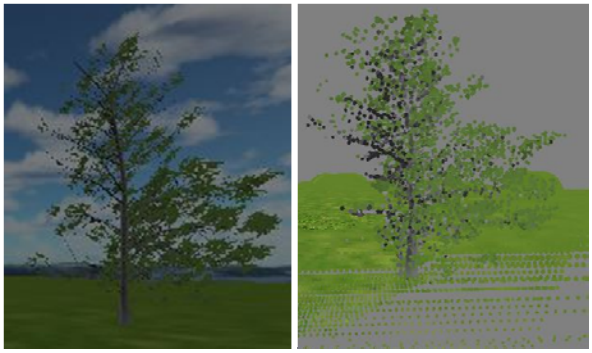

Figure 13: High Spatial Frequencies Rendered as Points. The colorized points at right suggest a tree for very little computational cost or modeling complexity relative to the detailed polygonal tree model shown to the left.

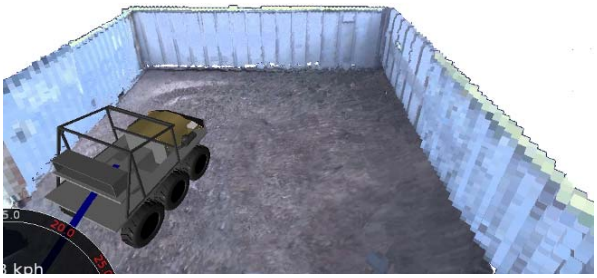

Figure 14: Legos. Projecting video onto small cubes at the correct range is an inexpensive way to visualize data whose surface shape cannot be resolved. Here, the vertical walls of trucking containers are very clear despite the inaccuracy of their local surface geometry.

5 Teleoperation Algorithms

So far, we have described the basic mechanisms for producing and rendering a photorealistic model that surrounds a moving vehicle. This section describes how this basic mechanism is augmented to produce a teleoperation system.

5.1 Motion Prediction

Any real wireless communications system will introduce latency into the telemetry passing between the vehicle and the OCS. It is well known that such latency is one factor that makes teleoperation difficult. In (Sheridan, 1993) the basic problem is explained in terms of the capacity of a delay to convert negative feedback to destabilizing positive feedback.

The capacity to render the vehicle from an external viewpoint not only provides hemispherical exterior context to the operator, but it also provides the opportunity to remove latency using prediction. The vehicle display is virtual anyway so it is straightforward to draw the vehicle anywhere on the display. We render the vehicle at its predicted position at the time in the future when commands issued by the operator will arrive at, and be acted upon, by the vehicle (Figure 16). This technique produces a continuously predictive display that appears to respond with no latency.

The predicted position of the vehicle is computed based on solving a velocity driven dynamics model. This problem is essentially the same as the problem of dead reckoning in 3D with the added component of determining response velocities given input velocities and terrain shape. Our solution is derived from the model of (Kelly & Stentz, 1998) which enforces a terrain contact constraint. This model was originally used for obstacle avoidance purposes where predictions many seconds into the future are required. While it does not attempt to model the forces of terrain contact and traction, it has been more than adequate for simulating forward by the sub-second delays that we experience. See Figure 15 for the basic data flow.

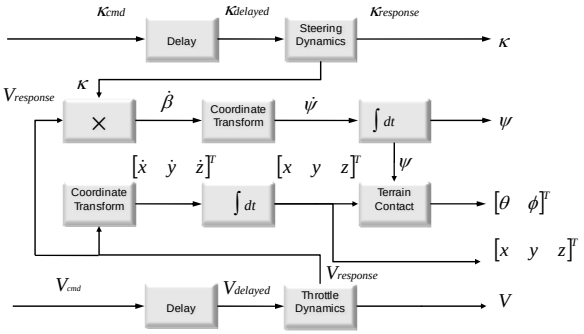

Figure 15: Motion Prediction: A velocity driven nonlinear vector differential equation is used to predict steering and speed response while remaining in contact with the terrain.

The command inputs are curvature κ and velocity V. Each of these sampled signals is passed through a FIFO queue to create a time delay. The delayed signals are then modified by calibrated models that account for such effects as input bounds, time constants, and wheel slip. The two response signals are multiplied to produce the angular velocity of the vehicle about the body frame z axis. The projection of this rate onto the world z axis is the yaw rate, whose integral is the yaw.

Likewise, the linear velocity is projected onto all 3 axes of the world frame and integrated to produce the updated position. Once the position and yaw are known, the pitch θ and roll ϕ angles are computed by allowing the vehicle to settle into a minimum energy pose determined by imaginary springs which connect the wheel contact points to the underlying terrain.

5.2 Latency Compensation

Teleoperation systems experience three types of latencies: input delays, output delays, and process delays. We typically measure time delay by using GPS time at both the vehicle and the OCS and by tagging all data packets with the GPS time. The OCS needs only to subtract the packet time tag from the present time to determine the input latency. The output delay at the present time cannot be measured, but that of earlier cycles can be computed on the vehicle and sent to the OCS for use in subsequent predictions.

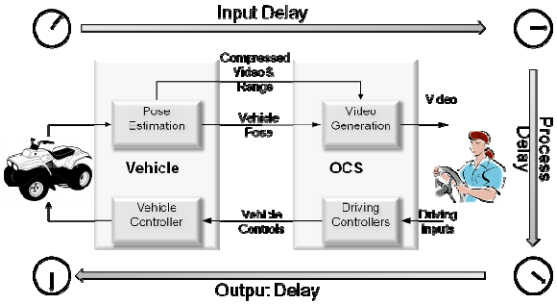

Figure 16: Latencies. Delays are introduced by communications in both directions as well as by the operator and processing at the OCS.

Latency in the video will still exist, of course, and it will be noticeable if objects are moving in the scene or on rough terrain. In the first case, objects other than the virtual vehicle will be in delayed position on screen. In the second, large range shadows behind hills will become more evident. Latency will also be noticeable if a vehicle with a narrow sensor field of view turns a sharp corner or passes an occluding object (like a building). In the first case, an overhead view would reveal that the video field of view is rotated backward in time with respect to the virtual vehicle. In the second case, the capacity to see around the corner of a building would be reduced relative to a true zero latency system.

This prediction technique also trades apparent latency for the effects of prediction error. If the prediction system is not adequate, the vehicle will appear to jump on screen as each new incoming pose is processed. The sensitivity of predicted position error to initial heading is high and proportional to both latency and velocity. Nonetheless, we have found that operators uniformly prefer using latency compensation to driving a system with latency. Presumably this is because the OCS is performing the time extrapolation and visualization that would otherwise have to be done intuitively by the operator. We have sometimes rendered both the delayed vehicle and the predicted one to make the process clearer. On such a display the two vehicles come together when the vehicle is stopped, and they separate during motion by a distance and angle proportional to the latency and linear and angular velocity.

5.3 Simulated Leader Follower

Given the capacity to predict the future, a potentially more useful technique is to predict slightly beyond the command arrival time to produce a display of a vehicle slightly into the future. In this case, some of the prediction error has not happened yet, and the real robot can be given the responsibility to reduce the error before it occurs. This is accomplished by considering the simulated vehicle to be a lead vehicle, which the real one is obliged to follow. In this case, the path followed by the simulated leader is passed to the real vehicle as the command input. Some operators prefer this simulated leader technique to basic motion prediction.

A more precise description of the distinction is as follows. In predictive display mode, the objective of the OCS in Figure 17 is to predict station 5 as close as possible to station 4. On the other hand, in simulated leader mode, the objective of the real vehicle (which sent pose data at station 1 but is now at station 4) is to drive from station 4 as close as possible to station 6. This simulated leader approach trades jitter of the predicted vehicle on the display for the negligible effect of slightly increasing the real latency between the on screen vehicle and the video it apparently produces.

In either case the simulated (predicted or lead) vehicle is rendered in the context of the 3D Video feed that is updated to include new information as the real (but not usually displayed) vehicle moves. Hence, the operator has the sensation that the on-screen vehicle is producing this video. In reality, it is being produced by a real (not displayed) vehicle somewhat behind it in time and space.

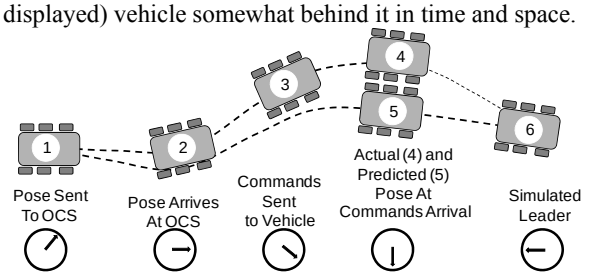

Figure 17: Prediction for Latency Compensation. Prediction of station 5 can be used for latency-free rendering. Errors in the initial state and prediction process may predict the vehicle will arrive at station 5 when it will really arrive at station 4. Such errors can be treated as path following errors if a simulated leader is rendered at station 6 thereby making the real vehicle responsible for compensation of prediction errors.

The state of the leader may or may not be bound to start prediction from the pose feedback coming from the real vehicle. If it is, prediction error may still cause jumps on the display. If not, the real vehicle becomes responsible for following the space-time trajectory of the leader, and extra mechanisms will need to be in place to deal with the case where the real vehicle is unable to keep up. When the real vehicle is too far behind the leader, the leader will be rendered into less accurate regions of the model that are produced from longer range sensor data. It is also possible to use both mechanisms at once by slowly biasing the simulated leader to be a fixed time from the predicted one based on the last pose received.

The simulated leader follower's performance depends on the capacity of vehicle control to reduce following error with respect to a virtual lead vehicle which is usually very near in space. In order to remain relevant to complex maneuvers in tight quarters, we use a simplified version of the controller described in (Howard, Green, & Kelly, 2009).

Despite the implementation issues described here, the above latency compensation techniques have proven to be very valuable in practice. In one experiment (Figure 18), for example, we demonstrated unprecedented high speed driving on dirt roads while avoiding discrete obstacles.

Figure 18: High Speed Obstacle Avoidance. Latency compensation is most valuable during high speed driving. Here, the operator avoids an obstacle by fitting the vehicle into a narrow space to its right. A custom fly-behind view was used. The speed reached approximately 24 km/hr. The operator control station is about 1 km farther down the road.

5.4 Telemetry Compression

Virtualized reality creates an opportunity to implement effective data compression in remote control applications. The fact that the rendering of the model is decoupled from the frame rate of the cameras means that the update of the display, showing a moving vehicle, can proceed at high rates regardless of the camera rates and the pose update rates. This capacity also implies a high degree of robustness to dropped frames of either type since each appears to be a momentary increase in input latency for which the system is already configured to compensate.

The input pose data and output control signals are small enough to be irrelevant to the compression issue. Lidar data is converted from floating point to 16 bit integers. For the video, we use an Xvid (MPEG-4) encoder on the vehicle and a decoder on the OCS to perform compression. Visible parts of the vehicle are cropped. Then the three streams from the sensor pods are reduced to 10Hz frame rate before they are compressed. Based on just these measures, we are able to produce very high quality displays that compete with full frame video using only 1 Megabit /sec of communication data rates (Figure 19).

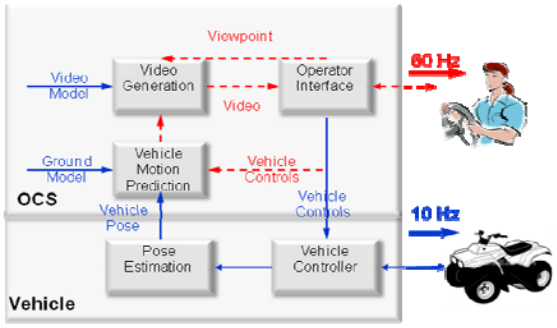

Figure 19: Telemetry Compression via Dropped Frames. The commitment to virtualize the entire scene makes it possible to update the display faster than the video, or to achieve compression by deliberately dropping frames.

5.5 Photorealistic Large Scale Mapping

While the 3D Video system is designed primarily for operator interface purposes, the construction of a photorealistic model is fundamental to its operation. Its capacity to remember a model of everything that has been seen in the immediate vicinity of the robot leads to a capacity to create large scale photorealistic maps provided that:

- offline memory capacity is adequate
- the data can be properly registered in space

We routinely equip our robots with enough disk space to store all of the raw data gathered in a day of operations, and an integrated model performs compression as a byproduct of its operations since it eliminates redundant measurements of the same scene point. Adequate disk space for large scale maps is therefore always available.

The revisiting problem of SLAM (Stewart, Ko, & Konolige, 2003) is that of recognizing perceptually that

the robot has returned to a previously visited location. Our large outdoor vehicle uses high accuracy INS-GPS localization so this problem is solved easily to centimeter accuracy by our state of the art localization system instead of by perception.

Figure 20 shows an overhead view of a map of a "maze" produced by arranging trucking containers during one of our other experiments. This exercise was designed to test the value of mapping while navigating a complex unknown environment. Operators driving the robot remotely were charged to find and classify objects while exploring the maze during the search.

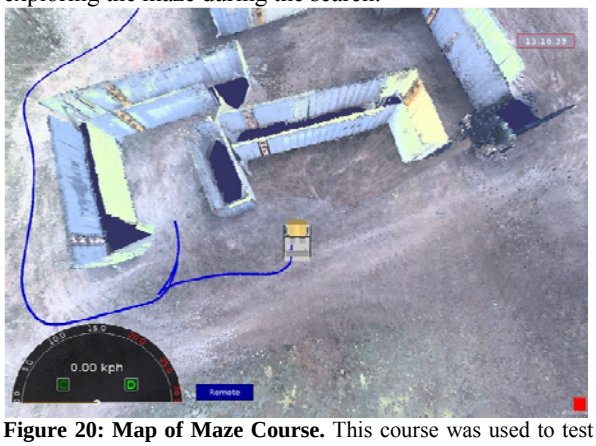

Figure 20: Map of Maze Course. This course was used to test the capacity of operators to solve a maze while conducting a search. This highly accurate photorealistic 3D model was produced from on-board perception and our visualization system.

5.6 Augmented Reality and Mixed Initiative Interactions

Both the point cube and the terrain map are standard components of our autonomy systems that are produced for the purpose of obstacle and hazard detection (Jakel, Krotkov, Perschbacher, Pippine, & Sullivan, 2006) (Kelly, et al., 2006). Given such algorithms, it is natural to wonder how they can be used to help the operator drive.

Figure 21 shows a simple augmented reality display where the classifications of simple slope-based obstacle detection algorithms are used to partially color the terrain. The colors are blended with the video such that reddish areas are to be avoided, and greenish ones are safe for driving.

In benign terrain, in broad daylight, this augmented reality facility may not add much value. However, when terrain is difficult or lighting or visibility is poor, such an autonomy system could add value if the human interface was configured correctly. Lidar works better at night due to reduced solar interference, and infrared appearance data can be processed like daytime video to produce a system that permits an operator to drive in total darkness.

The 3D Video system uses many of the same data structures for rendering and autonomy, so the operator and the autonomy system can interact more readily through the display; augmented reality is but one such mechanism. It is possible to have autonomy veto operator commands or bring the vehicle to stop, and the display can likely provide the operator with the reason for the robot initiative.

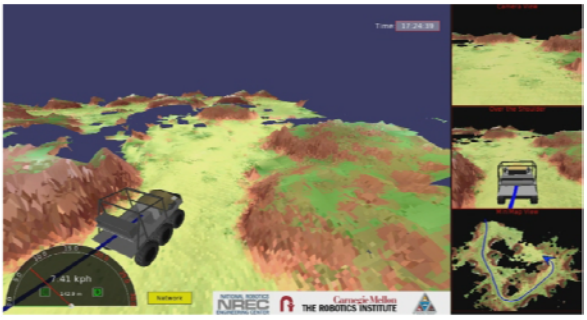

Figure 21: Augmented Reality Display for Autonomy Assisted Hazard Avoidance. The photorealistic display is augmented with false color obstacle annotations. Billboards are turned off.

6 User Study

The 3D Video system has been evaluated in a week-long field evaluation. The test focused on demonstrating measureable human interface improvements. Its results are discussed below.

6.1 Experimental Design

The goal of 3D Video technology in this experiment was to increase an operator's awareness of the surrounding context of the vehicle being controlled, thereby reducing operator errors and increasing the speed with which tasks could be completed.

We conducted an operator performance assessment involving five operators of different skill levels. The test was conducted in a barren, mostly flat outdoor area in Pittsburgh in December 2007. An earlier version of the vehicle described above was used. It had only one CR sensor so its field of view was more limited. The test course (Figure 22) was designed to elicit errors known to occur commonly in teleoperation such as collisions and unsafe path deviations.

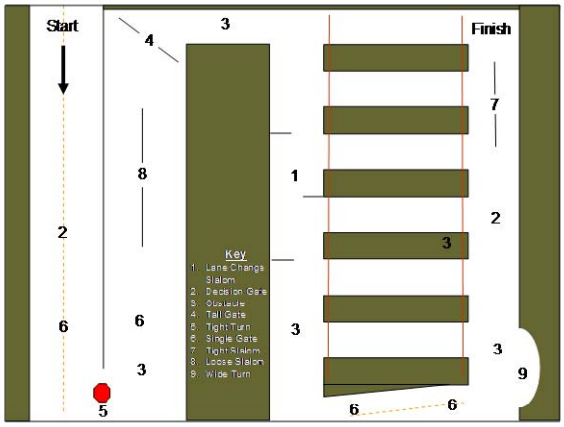

Figure 22: User Study Test Course. Specific maneuvers are indicated by number. 1 = lane change slalom, 2 = decision gate, 3 = obstacle, 4 = tall gate, 5 = tight turn, 6 = single gate, 7 = tight slalom, 8 = loose slalom, 9 = wide turn.

The participants averaged 20 years of automobile driving experience. Three subjects had prior experience teleoperating a live vehicle, including one with a 3D video system. Two of these subjects had participated in one other experiment, while the other had extensive experience teleoperating a vehicle in many experiments. Three subjects had minimal experience teleoperating a *simulated* vehicle (two of these included in the group with live vehicle experience). Four subjects had been playing driving-based video games for an average of 13 years, with one subject playing as often as a few times per week. One subject had never played a driving based video game.

Each operator drove the course in 4 different ways including sitting in the vehicle using standard controls, basic teleoperation with live video, and 3D video with and without latency compensation. Each driving mode was assigned in random sequence to remove bias associated with learning the course and the OCS.

Latency compensation refers to the predictive display (discussed earlier) that was used to alleviate the effects of video latency. Latency varied from 0.1 seconds to as high as 0.5 seconds throughout the test due to such uncontrollable factors as antenna occlusion and interference. Communications were limited to 1 megabit per second data rates uplink from the vehicle. Data transmitted down to the vehicle from the OCS was negligible.

The course consisted of a paved roadway with traffic cones set up to guide drivers at particularly ambiguous areas such as intersections. Course features included slaloms, decision gates, discrete obstacles and a series of loose and tight turns. Difficulty ranged from quite easy to quite difficult. The course was difficult enough to induce errors even when driving a vehicle manually from the in-vehicle driver's seat.

Performance metrics included course completion time, course accuracy, average speed and errors as well as subjective input on workload. We also asked the test subjects for impressions of the system, and recommendations for future improvement. Completion time and average speed are somewhat redundant when the duration of stops is removed because the distance of all tests is the same. Of course, the opposite treatment of stops is an equally valid and useful approach so both values are provided below in Table 1.

Errors were defined as hitting a cone, (having the vehicle emergency-stopped before) hitting an obstacle, or deviating from the defined region of the course (driving off the road). Obstacles were concrete barriers, fences, and hay bales that occurred sporadically along the perimeter of the course.

6.2 Experimental Results

Overall experimental results are summarized in Table 1 and discussed thereafter.

Metric	Live Video	3D Video	3D Video (latency compensated)	Manual Drive
Completion Time without Stop (min)	9.3 ± 1.0	7.2 ± 1.8	6.4 ± 0.8	2.2 ± 0
Completion Time with Stops (min)	10.4 ±1.5	7.8 ± 2.3	7.1 ± 1.4	2.2 ± 0
Average Speed (m/s)	1.0 ± 0.1	1.3 ± 0.4	1.5 ± 0.2	4.2 ± 0.5
Number of Stops	4.8 ± 3.3	2.8 ± 0.8	2.6 ± 3.0	0.0 ± 0
Errors	9.6 ± 4.2	5.0 ± 2.5	7.9 ± 3.2	2.4 ± 0.9
Workload (NASA TLX)	67 ± 12	54 ± 11	59 ± 8.0	59 ± 24

Table 1: User Study Summary Performance Metrics. Average performance for all operators in each driving mode is shown. Standard Deviations are written as tolerances.

A typical record of one test subject driving remotely is shown in Figure 23.

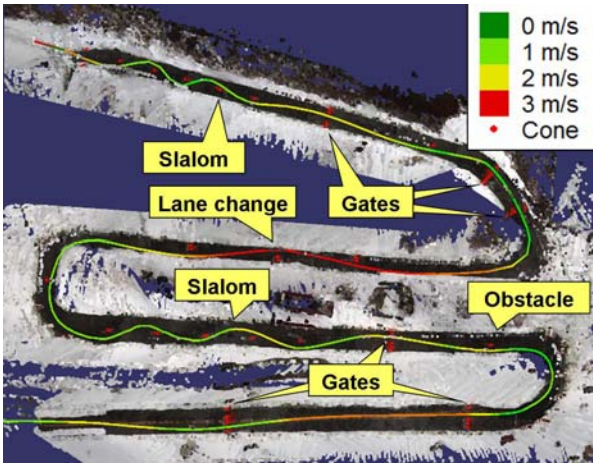

Figure 23: Typical Remote Control Test Result. This synthetic overhead view of the test range was produced by the 3D Video system for use in analyzing test results. Speeds are color coded.

6.2.1 Course Completion Time Results

3D Video enabled operators to complete the course faster than basic teleoperation. Completion times were approximately 20% lower with 3D Video alone and 30% lower when 3D Video was combined with latency compensation. As expected, manual driving (in the vehicle) was still far superior, with course completion time approximately 75% lower than basic teleoperation.

6.2.2 Speed Results

3D Video enabled operators to drive faster than basic teleoperation. Basic teleoperation achieved 1.0 m/s average speed, while 3D Video alone led to 30% faster driving, and 3D Video with latency compensation increased speed by 50%. Manual driving was almost three times faster than the best teleoperation.

6.2.3 Number of Stops

3D Video configurations reduced the frequency of stopping by 43% when compared to basic teleoperation. No drivers stopped during the manual driving configuration. The choice to stop the vehicle was a common response when relevant information was not available due to limited field of view (e.g. during turns) or because latency disoriented the operator.

6.2.4 Error Rate

3D Video reduced errors when compared to basic teleoperation. With 3D Video alone, the error rate dropped by almost 50%, while the error rated dropped by about 20% when 3D Video was combined with latency compensation. Manual driving was again the gold standard, with an error rate approximately 75% lower than basic teleoperation. The course was sufficiently complex that drivers did commit errors even with manual driving. The average rate while driving in-vehicle was 2.4 errors per run, and every driver committed at least one error over the course when driving in-vehicle.

Drivers made more errors with latency compensation than without. This effect was likely due to several nonidealities in the test including i) variable latency invalidating the constant-latency model used in the software, ii) sub-optimal vehicle model parameters for prediction, and iii) inaccuracies in the pose data. All of these effects contributed to errors in motion prediction that were at times substantial (relative to the tolerance of many of the course decision gates, for example). Constant latency was used due to problems in the GPS time tags that were subsequently fixed as evidenced by the result shown in Figure 18.

6.2.5 Workload

The NASA TLX workload questionnaire (Hart & Staveland, 1987) was administered after each run, allowing operators to rate perceived mental demand, physical demand, temporal demand, own performance, effort and frustration associated with each driving condition. Overall workload scores indicate the least amount of workload was required with the 3D visualization system alone. As expected, the highest workload was achieved with live video, while 3D Video with latency compensation and manual drive were rated similarly. In general, manual driving workload was rated higher than expected. This may be due to the physical effort required to use the vehicle steering wheel and a lower than anticipated perceived performance rating.

With a more detailed look at the components of overall workload, differences between driving conditions become more apparent. Live video required significantly more mental demand than other driving conditions, as well as higher temporal demand, perceived effort and frustration levels. Temporal demand ratings were very close. This is not surprising given that drivers were told to complete the course as quickly as possible. This goal created time-based workload across all conditions. 3D visualization conditions were rated similarly, but frustration levels were higher without latency compensation. Drivers reported the lowest physical demand with the 3D Video conditions.

6.2.6 *Exit Interview*

An exit interview was completed with each participant at the conclusion of all runs. The most commonly requested improvements for basic teleoperation include decreased latency, higher video frame rate and more cameras or unique viewpoints. Participants also mentioned better resolution, wider field of view, and an indication of vehicle position in the video frame that would allow them to drive through tight spaces. In general, operators wanted the ability to judge where the vehicle is positioned in the world by having a direct reference to all objects in the environment.

Participants felt that the greatest strength of the 3D Video was the vehicle model presented within the photorealistic model. The model made it easier to recover from mistakes, and it allowed operators to judge upcoming course events with respect to the vehicle. This capacity allowed them to respond to the environment more accurately. One operator commented: "I could go faster between events, and then slow down before an event. I could time the slow down better." 3D Video also provided a wider field of view, latency compensation, and selectable viewpoints. These features provided a "less stressful" environment and reduced the amount of time spent "paying attention to the vehicle," potentially freeing up time for other vehicle control and mission-related tasks.

3D Video improvement suggestions included reducing artifacts, a higher video frame rate, improvements in latency compensation, and a wider field of view for turns. A higher frame rate was suggested to make driving at a higher velocity easier.

The final portion of the exit interview allowed participants to rank their preferences for driving condition and 3D Video viewpoints. Manual driving was preferred, followed by 3D Video with latency compensation, 3D Video without, and live video. Three viewpoints were available within the 3D Video: native camera, over-the-shoulder, and overhead (bird's eye view). The overall preference for viewpoints was unanimous: over-the-shoulder followed by overhead and then native video (Figure 24).

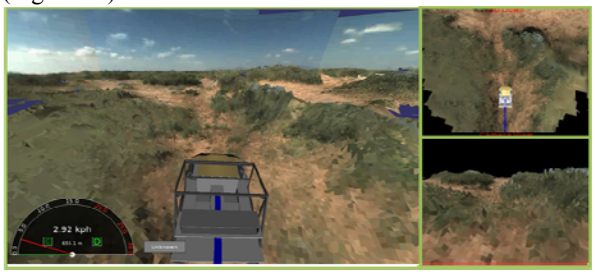

Figure 24: System Viewpoints. The operator could choose from one these three viewpoints with the other two reduced in size to the right. Left: Over the shoulder. Top right: Overhead. Bottom right: Native video.

Comments indicate bird's eye view was useful when navigating left or right for a short distance, such as in a slalom, and native location was useful if driving on straight roads for a long distance. Over the shoulder was more or less the "all purpose" preferred viewpoint.

7 Conclusions & Future Work

7.1 Future Work

The system presently makes a weak assumption of a static scene because the field of view is not omnidirectional. Our next sensing iteration will include an omnidirectional lidar and video system is order to support hemispherical situation awareness for the operator.

The system makes a stronger assumption when the data is remembered outside the sensor field of view, essentially, forever. Moving objects add an extra level of complexity worthy of significant study. Ideally these would be identified and removed from the model when too much time has passed to predict their positions accurately. Range data makes it possible, in principle, to disambiguate moving objects from the background and render the background when the region is outside the field of view of the perception sensors.

Our method does not directly model translucent, transparent, or porous objects (such as sparse vegetation). Typically, these objects are modeled based on the foreground object. For example, the scene behind a chain-link fence will be pasted onto the fence itself. While some work has been done on detecting layers in images, the current methods are not fast enough for real-time usage.

Finally, it should be possible to improve long-distance modeling using stereo or structure from motion, and we are presently investigating ways to fuse stereo and laser data for this purpose.

7.2 Conclusions

The skill level required for competent teleoperation in difficult situations is known to be substantial. However, most of us can quickly learn to drive a small remote controlled child's car from two joysticks given the benefit of a low latency interface and an external visual of the vehicle in the context of its surroundings. Hence the basic causes of the difficulty are the numerous nonidealities of the imaging, communication, and display devices commonly used in teleoperation.

This paper has proposed a method to convert the task of robot teleoperation into a synthetic form of line-of-sight

remote control. User studies have verified substantial gains in the effectiveness of the human-machine system. Along the way, we have produced improved solutions to problems like latency compensation and data compression for which there has been little hope of significant progress for some time.

While many component technologies have made this possible, the most novel is photogeometric sensing applied to virtualized reality. Photogeometric sensing has the capacity to produce displays with both the photorealism of video and the interactivity of a video game.

We expect that as sensors, rendering, guidance, and communications technologies continue to evolve, such displays and their derivatives will become a standard part of our toolbox. Technologies like flash lidar with bore-sighted video for ground vehicles will hopefully come on-line and reduce the engineering effort significantly. Even in the meantime, we find the effort is worthwhile in those very difficult applications for which robots and remote control are justified in the first place.

Our 3D Video system is basically a projective texturing engine added to visualize colorized range data sets that were already being produced for the purposes of autonomy. The mental model used by both operator and robot is virtually identical in our system, and this suggests many more derived advantages will be possible in contexts where autonomy shares more of the load, and human and robot cooperate more fully.

Acknowledgements

This work was funded by the US Army Tank Automotive Research, Development, and Engineering Command (TARDEC). The colorized ranging sensor concept described here was based on a concept originally developed under funding from John Deere Corporation. The autonomy work described in Figure 6 was funded by DARPA.

References

Anderson, D., & Kelly, A. (2005). Experimental Characterization of Commercial Flash Lidar Devices. *International Conference on Sensing Technologies.* Palmerston North, New Zealand.

Anderson, D., Howard, T., Apfelbaum, D., Herman, H., & Kelly, A. (2008). Coordinated Control and Range Imaging for Mobile Manipulation. *International Symposium on Experimental Robotics (ISER).*

Bejczy, A., Kim, W., & Venema, S. (1990). The Phantom Robot: Predictive Displays for Teleoperation with Time Delay. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).*

Buss, M., Peer, A., Schaub, T., Stefanov, N., & Unterhinninghofen, U. (2008). Multi-modal multi-user telepresence and teleaction system. *Intelligent Robots and Systems*, (pp. 4137–4138).

Chatila, R., Lacroix, S., Simion, T., & Herrb, M. (1995). Planetary exploration by a mobile robot: mission teleprogramming and autonomous navigation. *Autonomous Robots* .

Chen, S. E., & Williams, L. (1993). View interpolation for image synthesis. *SIGGRAPH 93*, (pp. 279-288).

Daily, M., Harris, J., Keirsey, D., Olin, D., Payton, D., Reiser, K., et al. (1988). Autonomous Cross Country Navigation with the ALV. *IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 718-726).

Debevec, P., Taylor, C., & Malik, J. (1996). Modeling and Rendering Architecture from Photographs. *SIGGRAPH*, (pp. 11-20).

Dima, C., Vandapel, N., & Hebert, M. (2004). Classifier Fusion for Outdoor Obstacle Detection. *International Conference on Robotics and Automation (ICRA)*, (pp. 665 - 671).

Edwards, L., Sims, M., Kunz, C., Lees, D., & Bowman, J. (2005). Photo-realistic Terrain Modeling and Visualization for Mars Exploration Rover Science Operations. *IEEE Systems, Man, and Cybernetics*, Waikoloa, Hawaii.

El-Hakim, S. F., Boulanger, P., Blais, F., & Beraldin, J. A. (1997). A system for indoor 3D mapping and virtual environments. *Videometrics V*, (pp. 21-35).

Ferland, F., Pomerleau, F., Le Dinh, C., & Michaud, F. (2009). Egocentric and exocentric teleoperation interface using real-time, 3D video projection. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*, (pp. 37-44).

Fong, T., Pangels, H., & Wettergreen, D. (1995). Operator Interfaces and Network-Based Participation for Dante II. *SAE 25th International Conference on Environmental Systems.* San Diego, CA.

Fong, T., Thorpe, C., & Baur, C. (2001). Advanced Interfaces for Vehicle Teleoperation: Collaborative Control, Sensor Fusion Displays, and Remote Driving Tools. *Autonomous Robots* , 77-85.

Früh, C., & Zakhor, A. (2004). An Automated Method for Large-Scale, Ground-Based City Model Acquisition. *International Journal of Computer Vision , 60* (1), 5-24.

Funda, J., Lindsay, T., & Paul, R. (1992). Teleprogramming: toward delay-invariant remote manipulation. *Presence: Teleoperation. Virtual Environonments , 1* (1), 29-44.

Hart, S. G., & Staveland, L. E. (1987). *Development of NASA-TLX (Task Load Index): Results of empirical*

and theoretical research (Vol. Human mental workload). (P. H. Meshkati, Ed.) Elsevier.

Hine, B., Hontalas, P., Fong, T., Piguet, L., Nygren, E., & Kline, A. (1995). VEVI: A Virtual Environment Teleoperations Interface for Planetary Exploration. *SAE 25th International Conference on Environmental Systems.* San Diego, CA.

Hine, B., Stocker, C., & Sims, M. (1994). The Application of Telepresence and Virtual Reality to Subsea Exploration. *2nd Workshop on Mobile Robots for Subsea Environments (Proc. ROV'94).* Monterey, CA.

Ho, N., & Jarvis, R. (2007). Large scale 3D environmental modelling for stereoscopic walk-through visualisation. *3DTV07*, (pp. 1-4).

Hoiem, A., Efros, A., & Hebert, M. (2005). Geometric context from a single image. *International Conference on Computer Vision (ICCV),*, (pp. 654–661).

Hu, S. Y., & Neumann, U. (2003). Approaches to large-scale urban modeling. *IEEE Computer Graphics and Applications (CG&A) , 23* (6), 62-69.

Huber, D., Herman, H., Kelly, A., Rander, P., & Warner, R. (2009). Real-time Photorealistic Visualization of 3D Environments for Enhanced Teleoperation of Vehicles. *Proceedings of the 2nd International Conference on 3D Digital Imaging and Modeling.* Kyoto, Japan.

Jakel, L., Krotkov, E., Perschbacher, M., Pippine, J., & Sullivan, C. (2006). The DARPA LAGR Program: Goals, challenges, methodology, and phase I results. *Journal of Field Robotics , 23*, 945–973.

Johntson, J., Alberts, J., Berkemeier, M., & Edwards, J. (2008). Manipulator Autonomy for EOD Robots. *26th Army Science Conference.*

Kanade, T., Rander, P., & Narayanan, P. J. (1997). Virtualized Reality: Constructing Virtual Worlds from Real Scenes. *IEEE MultiMedia , 4* (1).

Kelly, A., & Stentz, A. (1998). Rough Terrain Autonomous Mobility, Part 2: An Active Vision, Predictive Control Approach. *Autonomous Robots , 5*, 163-198.

Kelly, A., Capstick, E., Huber, D., Herman, H., Rander, P., & Warner, R. (2009). Real-time photorealistic virtualized reality interface for remote mobile robot control. *Intl. Symp. On Robotics Research (ISRR),.* Lucerne, Switzerland.

Kelly, A., Stentz, A., Amidi, O., Bode, M., Bradley, D., Diaz-Calderone, A., et al. (2006). Toward Reliable Off-Road Autonomous Vehicles Operating in Challenging Environments. *The International Journal of Robotics Research , 25* (5-6), 449-483.

Kim, W. S. (1993). Advanced Teleoperation, Graphics Aids, and Application to Time Delay Environments. *1st Industrial Virtual Reality Show and Conference (IVR '93)* , (pp. 23-25). Makuhari Meese, Japan.

Kim, W. S. (1996). Virtual Reality Calibration and Preview / Predictive Displays for Telerobotics. *Presence: Teleoperators and Virtual Environments , 5* (2), 173-190.

Lacase, A., Murphy, K., & DelGiorno, M. (2002). Autonomous mobility for the Demo III experimental unmanned vehicles. *Assoc. for Unmanned Vehicle Systems Int. Conf. on Unmanned Vehicles (AUVSI 02).*

Lewis, R. A., & Johnston, A. R. (1977). A Scanning Laser Rangefinder for A Robotic Vehicle. *5th. International Joint Conference on Artificial Intelligence.*

Lloyd, J. E., Beis, J. S., Pai, D. K., & Lowe, D. G. (1997). Modelbased telerobotics with vision. *International Conference on Robotics and Automation (ICRA)*, (pp. 1297–1304).

Matusik, W., Beuhler, C., Raskar, R., Gortler, L., & McMillan, L. (2000). Image-Based Visual Hulls. *SIGGRAPH*, (pp. 369-374).

McMillan, L., & Bishop, G. (1995). Plenoptic modeling: An image-based rendering system. *Proceedings of ACM SIGGRAPH*, (pp. 39–46).

Messuri, D. A., & Klein, C. A. (1985). Automatic Body Regulation for Maintaining Stability of a Legged Vehicle during Terrain Locomotion. *IEEE Journal of Robotics and Automation* , 132-141.

Milgram, P., Yin, S., & Grodski, J. (1997). An Augmented Reality Based Teleoperation Interface For Unstructured Environments. *American Nuclear Society (ANS) 7th Topical Meeting on Robotics and Remote Systems*, (pp. 966-973).

Milgram, P., Zhai, S., Drascic, D., & Grodski, J. (1993). Applications of augmented reality for human-robot communication. *International Conference on Intelligent Robots and Systems (IROS)*, (pp. 1467-1472). Yokohama, Japan.

Möller, T., Kraft, H., Frey, J., Albrech, M., & Lange, R. (2005). Robust 3D measurement with pmd sensors. *Proceedings of the First Range Imaging Research Dat at ETH Zurich.*

Mordohai, P., Frahm, J. M., & Akbarzadeh, A. (2007). Real-time video-based reconstruction of urban environments. *ISPRS Workshop on 3D Virtual Reconstruction and Visualization of Complex Architectures.*

Nielsen, C. W., Goodrich, M. A., & Ricks, B (2007). Ecological Interfaces for Improving Mobile Robot Teleoperation. *IEEE Transactions on Robotics and Automation*. 23 (5), pp. 927-941.

Rander, P. (1998). *A Multi-Camera Method for 3D Digitization of Dynamic, Real-World Events, PhD thesis.* Pittsburgh, Pa: Robotics Institute, Carnegie Mellon University.

Ricks, B., Nielsen, C., & Goodrich, M. (2004). Ecological Displays for Robot Interaction: A New Perspective.

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), (pp. 2855- 2860).

Rohlf, J., & Helman, J. (1994). IRIS Performer: A high performance multiprocessing toolkit for real-time 3D graphics. *In Proceedings of SIGGRAPH*, (pp. 381–394).

Ross, B., Bares, J., Stager, D., Jacker, L., & Perschbacher, M. (2008). An Advanced Teleoperation Testbed. *Field and Service Robotics*. Springer.

Ryde, J., & Hu, H. (2008). 3D Laser Range Scanner with Hemispherical Field of View for Robot Navigation. *Proceedings of the 2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. Xi'an, China.

Schneider, M., & Klein, R. (2008). Enhancing textured digital elevation models using photographs. *Intl. Symp. on 3D Data Processing, Visualization, and Transmission (3DPVT)*, (pp. 261–268).

Se, S., & Jasiobedzki, P. (2008). Stereo-vision based 3D modeling and localization for unmanned vehicles. *International Journal of Intelligent Control and Systems , 13* (1), 47-58.

Segal, M., Korobkin, C., van Widenfelt, R., Foran, J., & Haeberli, P. (1992). Fast shadows and lighting effects using texture mapping. *SIGGRAPH*, (pp. 249-252).

Sheridan, T. (1986). Human Supervisory Control of Robot Systems. *Proceedings of the IEEE International Conference on Robotics and Automation*.

Sheridan, T. (1993). Space teleoperation through time delay: Review and prognosis. *IEEE Trans. Robotics Automation , 9*, 592-606.

Smallman, H., & St. John, M. (2005). Naive realism: Misplaced faith in realistic displays. *Ergonomics in Design*, Summer, 2005 (pp. 6-13).

Stamos, I., & Allen, P. K. (2002). Geometry and texture recovery of scenes of large scale. *Computer Vision and Image Understanding (CVIU) , 88* (2), 94-118.

Stewart, B., Ko, J., & Konolige, K. (2003). The Revisiting Problem in Mobile Robot Map Building: A Hierarchical Bayesian Approach. *Conference on Uncertainty in Artificial Intelligence*.

Theunissen, E., Koeners, G., Roefs, F., Rademaker, R., Jinkins, R., Etherington, T. (2005). Guidance, Situation Awareness and Integrity Monitoring with an SVS+EVS. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California.

Wellington, C., & Stentz, A. (2004). Online adaptive rough-terrain navigation in vegetation. *IEEE Int. Conf. on Robotics and Automation (ICRA)*.

Wellington, C., Courville, A., & Stentz, A. (2006). A generative model of terrain for autonomous navigation in vegetation. *Intl. Journal of Robotics Research (IJRR), , 25* (1), 1287–1304.

Westover, L. (1990). Footprint Evaluation for Volume Rendering. In Computer Graphics. *Proceedings of SIGGRAPH*, (pp. 367–376).

Williams, L. (1978). Casting curved shadows on curved surfaces. *Proceedings of ACM SIGGRAPH* , 270-274.