3DIM #****

Real-time Photo-realistic Visualization of 3D Environments for Enhanced Tele-operation of Vehicles

Anonymous 3DIM submission

Paper ID ****

Abstract

This paper describes a method for creating photo-realistic three-dimensional (3D) models of real-world environments in real-time for the purpose of improving and extending the capabilities of vehicle tele-operation. Our approach utilizes the combined data from a laser scanner (for modeling 3D geometry) and a video camera (for modeling surface appearance). The sensors are mounted on a moving vehicle platform, and a photo-realistic 3D model of the vehicle's environment is generated and displayed to the remote operator in real time. Our model consists of three main components: a textured ground surface, textured or colorized non-ground objects, and a textured background for representing regions beyond the laser scanner's sensing horizon. Our approach enables many unique capabilities for vehicle tele-operation, including viewing the scene from virtual viewpoints (e.g., behind the vehicle or top down), seamless augmentation of the environment with digital objects, and improved robustness to transmission latencies and data dropouts.

1. Introduction

The advent of relatively low-cost laser scanners has enabled the accurate geometric modeling of three-dimensional (3D) environments for various purposes. The addition of imagery from a digital camera enables photo-realistic models that can be used for visualization or analysis. For example, city modeling applications often involve mounting laser scanners and cameras on a moving vehicle to create realistic 3D models of urban environments [2, 6, 4]. These models are created in a relatively time-consuming off-line procedure, rather than in real-time, due to the computational demands of the process.

In this paper, we approach the problem of 3D environment modeling from the opposite direction, focusing on creating a realistic 3D model online and in real-time rather than as an off-line batch process. Such an approach has immediate and obvious applications for tele-presence and tele-operation. Our focus is on the benefits and improved capabilities that this approach offers for the tele-operation of vehicles in outdoor environments. Conventional vehicle tele-operation works by transmitting one or more video feeds from the vehicle to a remote operator. The limitations of this method make vehicle tele-operation a challenging task. Cameras have a limited field of view, so operators must navigate with minimal peripheral vision. Furthermore, once an object leaves the the field of view, the operator must rely on his memory and motion perception to estimate its location. 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. Video-based tele-operation is susceptible to data dropouts and latency. If the transmission link between the vehicle and operator is interrupted, the operator has no visual or positional feedback from the vehicle. Even without dropouts, high-latencies make steering difficult, since the control decisions must be made using outdated information.

Figure 1. Our approach models 3D environments realistically and in real-time. The sensors used to capture this scene are mounted on the vehicle, which is shown using a synthetic representation, but we render the scene from an over-the-shoulder viewpoint to improve situational awareness for the tele-operator, who is avoiding obstacles while driving at 23 kph.

3DIM
#****

(a) (b) (c)

Figure 2. The video-ranging module (a) consists of a video camera and a nodding laser range-finder. The Gator (b) and LandTamer 2 (c) vehicles were retro-fitted for tele-operation and used in our experiments.

Our approach mitigates the problems of video-based tele-operation by creating, in real-time, a photo-realistic 3D model of the environment surrounding the vehicle (Figure 1). The 3D model provides a natural scaffolding for storing visual information that is outside of the current camera field of view. Since the scale of the scene is known, it is possible to augment the scene with virtual objects, such as the vehicle itself, to provide spatial context for objects outside the camera field of view. With our approach, data dropouts prevent the model from being updated, but they don't prevent the operator from seeing the current environment model. It may be safe to continue driving for a short time even without data updates, enabling seamless bridging of temporary data interruptions. Long transmission latencies can be addressed by showing the operator the predicted vehicle location rather than its last reported location.

A real-time environment model also enables new user interface concepts that are not possible with video-based tele-operation. The scene can be viewed from an arbitrary viewpoint, allowing operators to tailor the viewpoint to specific tasks. An "over-the-shoulder" viewpoint from behind the vehicle may be best for general driving, while a top-down view is better for parking. Objects in the environment can be analyzed geometrically, for example, to determine if a path between obstacles is wide enough to pass through. Since the visualization process is separate from the environment modeling process, these tasks can run independently, allowing unique capabilities, such as fast update of the visualization but slower model updates based on available bandwidth, and simultaneous visualization of multiple virtual camera viewpoints (e.g., forward view, over the shoulder view, and top-down view).

2. Sensor and vehicle platform

Real-time 3D modeling from a moving vehicle depends critically on a good sensor and platform design. In our approach, we use a custom-built, self-contained sensor, known as the video-ranging module (VRM), which produces time-stamped images and 3D point data in the sensor's local coordinate system (2a). The VRM consists of a SICK LMS-291 laser scanner and a Point Grey Bumblebee 2 video camera. The laser scanner is a single line scanner that is mechanically actuated to nod up and down. The scanner has the following characteristics: maximum range – 80m, field of view – 90° H by 50° V (+10° to -40°), resolution – 0.5° horizontally, and data rate – 13,575 points/second. This sensor offers a good trade-off between cost, accuracy, maximum range, and data rate. The stereo capabilities of the camera are not used in this work, and while it would be possible to accomplish 3D modeling with stereo (e.g., as in [7]), the accuracy of current stereo algorithms is not as good as laser scanners. The camera provides 720 x 500 pixel images, with a 60 degree field of view at a frequency of 5 Hz. 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 VRM with respect to the vehicle frame is also estimated to allow sensor data to be transformed into world coordinates while the vehicle is moving.

We have conducted experiments using two different tele-operated vehicles. The first vehicle is based on a Gator platform from John Deere that has been retro-fitted to allow remote steering and throttle control (Figure 2b). The vehicle is also equipped with an inertial navigation system (INS) for estimating the vehicle pose, wireless communication, and on-board computers for vehicle control and data logging. The second vehicle is based on a LandTamer 2 platform, which was similarly retro-fitted for tele-operation (Figure 2c). The Gator vehicle is equipped with a single forward-looking VRM, while the LandTamer platform has three VRMs angled at 30° with respect to one another to

Figure 3. The operator control station allows the remote operator to steer the vehicle and control its speed while visualizing the environment from user-selectable viewpoints.

provide a panoramic active viewing region. The camera and laser data from each VRM, as well as the vehicle pose information, is time-tagged, compressed, and transmitted to the operator control station, where the model is constructed and visualized.

The operator control station consists of an off-the-shelf personal computer (Intel Q6600 quad-core 2.4 GHz CPU, GeForce 8800 Ultra video, and 4 GB memory) and monitor. It is equipped with a steering wheel and pedals (Logitech MOMO) for controlling the vehicle (Figure 3). Additional buttons and controls on the steering wheel panel allow the operator to control the primary viewpoint, vehicle direction (forward or reverse), and cruise control.

3. 3D visualization overview

Given a laser scanner mounted on a vehicle as described in Section 2, it is relatively straightforward to colorize the

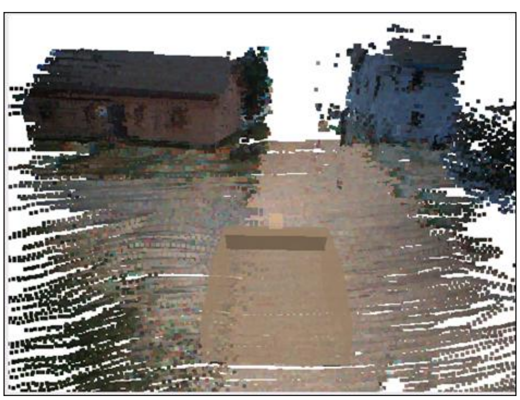


Figure 4. Simply colorizing the points from the laser scanner produces an unsatisfactory visualization. Objects are blurry, and gaps appear between the points, especially in regions seen close-up.

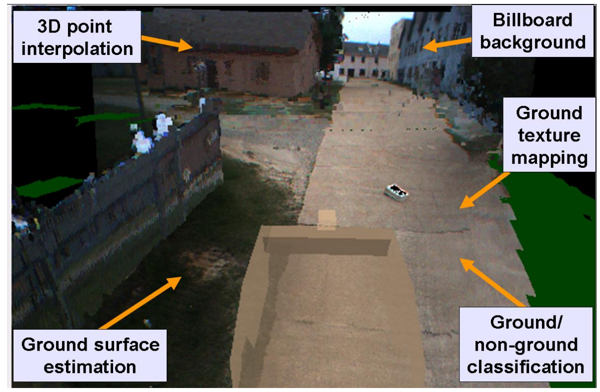


Figure 5. Our approach combines several modeling techniques, including estimating and texturing the ground surface, detecting and modeling non-ground surfaces, and filling in distant regions with a panoramic billboard.

points by projecting them into the most recent camera image and then display the resulting colorized point cloud in real-time. Unfortunately, the results, as shown in Figure 4, are not photo-realistic, and it would be difficult to tele-operate a vehicle using just this information.

There are a couple of significant problems with this naïve implementation of real-time environment modeling. First, the laser data is at a much lower resolution than the image data – angular resolution of 1° for the laser versus 0.083° for the camera. As a result, more than 99% of the image information is being discarded. Second, the laser data has a limited range, which is shorter than what is needed to tele-operate a vehicle even at moderate speeds. For road surfaces, the effective maximum range is about 25 meters, less if the road surface is dark or wet. Beyond this distance, the road shape, as well as obstacles in the road, are completely unknown to the operator, which necessitates driving at slower speeds.

Our approach addresses the limitations of the naïve approach through several independent techniques, which, when taken together, result in a photo-realistic environment model that enhances the tele-operation experience (Figure 5). The environment is divided into three classes of information to be modeled: ground surfaces, non-ground surfaces, and distant regions. We explicitly model the ground surface as an elevation map and apply a texture map to that surface using the video imagery. We use two different approaches to non-ground surface modeling. One technique is to interpolate the raw point data to be approximately the same resolution as the image data and then use the same colorization method that is used for the raw 3D points. The second technique is to model the objects using solid voxels wherever the data is sufficiently dense and then texture map these voxel surfaces. Finally, regions beyond the 3D sensor horizon are modeled by a planar "billboard" geometry that

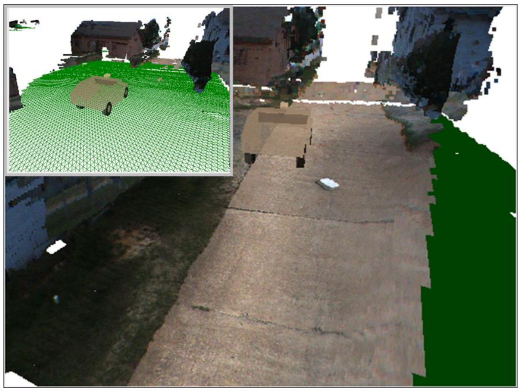


Figure 6. The ground surface is estimated using an elevation map, triangulated (inset), and texture mapped. The texture extends behind the vehicle, outside the current sensor field of view, giving the operator historical context.

is visually realistic as long as the virtual viewpoint is not too far from the actual camera. These techniques are detailed in the next several sections.

4. Ground surface modeling

The ground surface is modeled using a gridded height map with square cells. The process of estimating the ground surface height is relatively straightforward provided we know which 3D points are part of the ground and which are not. Conversely, the classification of points as ground or non-ground is simplified if we have an estimate of the ground height. While there are sophisticated ways to approach this "chicken-and-egg" problem through probabilistic reasoning or iterative optimization methods (e.g., [15]), the real-time constraints necessitate a more computationally tractable approach. In our approach, we first estimate ground heights based on all the points, and then use this height estimate to classify points as ground or non-ground.

As points are received, they are transformed into world coordinates using the current vehicle pose. The points are added to a sparse 3D grid of voxels, which is kept in world coordinates. The ground surface is represented by a 2D elevation map coincident with the x-y plane of the voxel grid. The height of a cell in the elevation map is set to the average height of the points found in the bottom-most occupied voxel in the column of voxels above that cell.

Because of the limited resolution of the 3D sensor, it is possible that some cells in the height map may have no measurements. We therefore apply an interpolation algorithm to fill small holes in the height map. We use linear interpolation across gaps smaller than a configurable number of cells, first computing in the X direction and then in the Y direction for any remaining holes. We also experimented with more complex methods, such as Kriging, but we found that the visual improvement was not enough to offset the added computational cost.

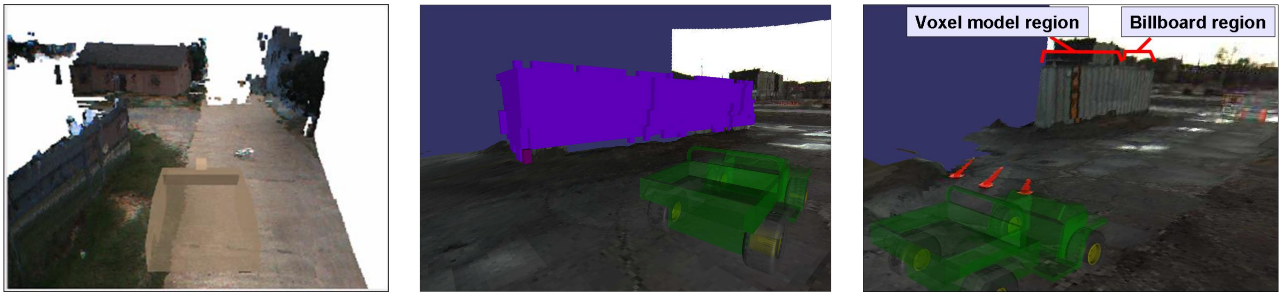
Next, the height map is triangulated (Figure 6, inset). While it is possible to triangulate using the height map cell centers directly, we first estimate the heights of the cell corners by averaging the heights of the occupied neighbors. This extra step allows triangulated geometry to extend to the very edge of the height map rather than leaving a half-cell of empty space that would have to be handled as a special case.

Finally, the ground surface texture is computed (Figure 6). We have experimented with two different texture mapping techniques. The first method, described here, creates an explicit texture map for the ground surface using a method similar to that used in [11], while the second method, described in Section 7, uses the image itself as a texture map. For explicit texture mapping, the 2D corners of the elevation map cells are mapped onto a blank texture map uniformly, so that each elevation map cell is allocated the same number of pixels in the texture map. For each 3D triangle in the ground surface, the source texture is found by projecting the triangle into the latest camera image. The destination texture is determined by the aforementioned mapping of elevation map cells onto the texture map. The source image triangle is then warped into the shape of the destination triangle in the texture map, and the image data is copied using bilinear interpolation. Note that this is an approximation to the true interpolation that a projective warping would need, but the effect is negligible for small triangles. The texture for ground surface triangles that project outside of the camera's field of view are not updated. As a result, the ground surface maintains the texture of the last known view of that surface patch, and a history of the ground surface behind the vehicle is preserved.

For efficiency, the ground surface height map is divided into tiles consisting of a fixed number of cells in each direction. This ground surface tiling allows us to efficiently handle an arbitrarily large ground surface map. As the vehicle travels, regions that are beyond the sensor range and outside the area that the operator needs for driving can be saved to disk or discarded. Only those tiles where new data is obtained need to be recomputed in a given time step, which eliminates redundant computation on unchanging map regions.

5. Non-ground modeling

Non-ground points are processed differently from ground points, since they cannot be modeled as a height map. Non-ground points are segmented from ground points using a threshold-based classification strategy. Any point more than a threshold distance above the currently estimated ground surface are considered to be non-ground points. The threshold is set based on the laser scan-



(a) (b) (c)

Figure 7. We developed two methods for modeling non-ground regions. Non-ground points can be interpolated to provide dense, colorized point clouds (a), or they can be modeled using solid voxels (b), which appear realistic when texture-mapped (c).

ner uncertainty and pose uncertainty. One side effect of this method is that non-ground points very close to the ground are not modeled, causing non-ground objects to float slightly above the ground. This effect is not very noticeable, however, and it could be addressed by a more complex classification algorithm.

We have developed two methods for modeling non-ground regions. The first method is the simplest. The non-ground points are simply interpolated (Figure 7a). Since the 3D point measurements are regularly sampled in both azimuth and elevation, groups of four neighboring points form a quadrilateral in 3D. New points are created by bilinearly interpolating between these four points. A surface orientation check is performed to prevent interpolation across obliquely viewed quads (which are likely to be depth discontinuities in the scene). Once the points are interpolated, they are colorized by projecting them into the current camera image. Point projection efficiency is improved by dewarping the camera image to remove lens distortions. The interpolation resolution can be set dynamically to match the image pixel resolution.

The second method for non-ground modeling represents non-ground objects using sets of occupied voxels (Figure 7b,c). An occupied voxel is formed whenever the density of points within a single voxel reaches a certain threshold (essentially a simplified 3D occupancy grid [10]). A region-growing algorithm is used to group occupied voxels into objects based on physical adjacency in the 3D grid. The voxel-based objects can be colored based on their object identity (e.g., object one is colored blue, object two is colored red, etc.), or the outer surfaces can be texture mapped. We use the projective texturing method described in Section 7 for this purpose.

6. Distant surface modeling

Surfaces beyond the range of the laser scanner have no geometry associated with them. For those situations, the environment is modeled using a planar billboard surface. In the simplest case, the billboard is a projection of the camera image (with lens distortions removed) onto a planar surface located in the environment (Figure 8). The billboard surface must be properly positioned with respect to the camera. That is, the surface must be parallel to the actual camera imaging plane, located along the camera's optical axis, and sized adequately to encompass the reprojected image. Unlike the geometry of the other aspects of the environment model, the billboard moves with the vehicle, or more precisely, with the camera.

A billboard gives a realistic visualization of the distant objects in the scene, such as the sky, distant road surfaces, and buildings. When the virtual camera is at the same position as the actual camera, the transition between the modeled geometry from the ground and non-ground objects and the unmodeled geometry in the billboard is almost imperceptible. As the virtual camera moves away from the actual camera position, the distortion becomes more apparent. However, since the unmodeled surfaces are relatively far away, the effect is realistic even for fairly large differences between virtual and real camera position.

One challenge of 3D modeling of complex scenes is ensuring that objects in the environment are only modeled once. For example, a non-ground object, such as a tree, should not also appear in the billboard background. Such duplicate objects can be confusing and detract from the realism of the model. Our approach to this problem is to filter out these foreground objects from the billboard background by making such regions transparent. We use a method based on the well-known technique of shadow-mapping [16]. The scene is rendered from the perspective of the latest camera image to determine the scene depth from that viewpoint. Then, when rendering the scene from a virtual viewpoint, a test is performed to determine whether a given rendered pixel on the billboard should be textured or transparent. The test checks whether the depth of the billboard point (as seen from the real camera's viewpoint) is greater than the depth of the closest object along that same ray. If so, then the point is occluded and should be transparent, otherwise it

3DIM #****

(a) (b) (c)

Figure 8. A panoramic billboard background fills in the distant regions of the scene. When viewed from the side (a), the billboard can be seen floating a fixed distance in front of the vehicle. When viewed from locations closer to the vehicle, the transition from 3D to planar billboard is nearly invisible (b), and distortions are not significant even at fairly large distances from the true camera viewpoint (c).

should be textured with the corresponding information from the camera image. This method is efficiently implemented using OpenGL and GPU programming.

7. Texture-mapping using projective texturing

As an alternative to explicitly texturing surfaces as described in Section 4, we also experimented with a more efficient and visually realistic texturing method using projective texture mapping [13]. Projective texture mapping textures a scene as if the texture map were projected onto the scene by a slide projector. In our case, the camera image (with distortion removed) is projectively textured onto the scene. When the virtual camera viewpoint is similar to the real camera viewpoint, this approach uses the texture map pixels very efficiently, since small, far away regions in the scene correspond to small regions of the camera image, while large, nearby scene regions correspond to large regions of the image. In such cases, rescaling of the original texture is minimized, preserving the detail of the original image to the extent possible.

We implemented projective texture mapping using GPU programming. The baseline approach is to texture using the most recent camera image. However, this method does not preserve scene regions once they pass outside of the camera field of view. To address this limitation, we maintain a history of previous camera images and use multiple textures. The number of previous images is limited by the texture memory of a given graphics card as well as limits on the argument list length of GPU shader routines. We have found that with current graphics cards, a maximum of 12 images is typical. Rather than use the last N images from a given camera, which would be highly redundant, our strategy is to selectively pick images spaced evenly over a given distance of vehicle travel (e.g., every 2 meters). The result, as shown in figure 9, is a realistic textured history extending a fixed distance behind the vehicle. The textures must be applied in temporal order to ensure that the latest view of a particular surface is displayed. The disadvantage of projective texture mapping is that the historical texture map cannot extend indefinitely. We are investigating methods to preserve the textures by transferring the texture information from the GPU back to the CPU or by combining the projective texture mapping method with the manual texture mapping method described in Section 4.

Figure 9. Projective texturing is a hardware-based alternative to texture-mapping that improves realism and efficiency.

8. Results

We conducted a formal user study using the Gator vehicle platform to compare the performance of our 3D tele-operation approach to video-based tele-operation and to direct driving (i.e., driving the vehicle normally). The task was to drive the vehicle on a predetermined route through a challenging obstacle course consisting of several narrow gates, sharp turns, lane-changes, and slaloms (Figure 10). Five users with varying skill levels navigated the course under each driving condition (direct driving, video-based tele-operation, and 3D-based tele-operation). The users were first trained using a separate training course to familiarize themselves with the vehicle and system capabilities. The user trials were randomized to limit the effect of experience on the test course from previous trials. The results

3DIM #****

(a) (b) (c)

Figure 10. In a user study, subjects tele-operated a vehicle through an obstacle course (a). The view from the single on-board camera provides limited knowledge about the vehicle's position relative to the obstacles (b), while the virtual viewpoint in our 3D model provides the needed context to localize the vehicle.

of the study indicate that the 3D tele-operation approach significantly improves performance, both in terms of driving speed and reduced number of errors (i.e., obstacles hit) when compared to video-based tele-operation. On average, driving speed was 30% higher using our proposed approach (1.3 kph vs. 1.0 kph) and had 48% fewer errors (5.0 vs. 9.6). Users uniformly reported that they preferred the 3D-based tele-operation mode to the video-based mode. Subsequent informal, experiments further validated the approach, which allowed tele-operation at speeds of up to 25 kph on dirt roads with obstacles (1).

User studies have demonstrated the benefits and unique capabilities that 3D environment modeling offers for tele-operation. One of the key advantages of the approach is the ability to view the scene from arbitrary viewpoints, rather than being limited to the original camera viewpoint. One viewpoint that is particularly useful for tele-operation is the "over the shoulder" view, in which the virtual camera is placed behind and slightly above the vehicle. This viewpoint, which is common in driving video games, allows the operator to see the vehicle (or an augmented reality version of the vehicle) and its relationship with the environment. Obstacles that have passed outside the camera's field of view can still be seen, and steering decisions can be made accordingly. This capability allows an operator to safely navigate tight spaces that would not be possible with video-based tele-operation. A top-down viewpoint is also useful navigating tight spaces and also for tasks like parallel parking and driving in reverse (Figure 11).

9. Related work

The ideas presented in this paper are closely related to research in several related fields, including 3D computer vision, computer graphics and rendering, tele-presence and augmented reality, and autonomous robots.

Most similar in spirit to our work is that of Johnston et al, who have developed a real-time method that uses stereo imagery and ladar to visualize a 3D environment for tele-operating a manipulator [7]. Their method uses either colorized points (similar to our baseline setup), textured triangles, or quads. Our approach differs in the details of the implementation, and our method handles both near-field and far field scene elements.

Modeling using laser scanners and imagery has been well-studied, especially in the context of modeling urban environments from terrestrial sensors [2, 6, 4]. Other work focuses on individual buildings or terrain models [14, 12, 1]. These systems, while able to produce photo-realistic models, work off-line using batch data.

Various methods for generating virtual viewpoints of a scene have been developed over the years. Image-based rendering techniques allow novel views to be synthesized using images only, but the methods are limited to viewpoints close to or between camera viewpoints [9]. Camera-based methods can also be used to create 3D models, using methods such as virtualized reality [8]. Recent work has shown that 3D can be extracted from a single image [5], but these methods are not as accurate as laser scanners and do not work in

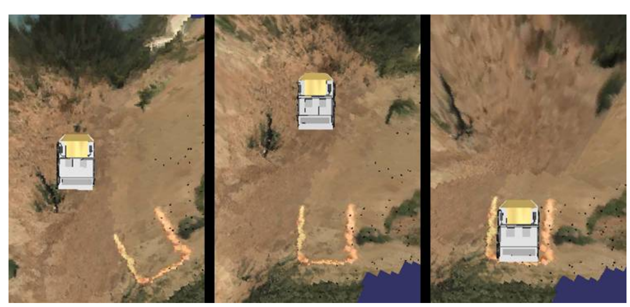


Figure 11. An overhead view allows task-specific operations, such as reversing the vehicle into a parking space without the benefit of a rear-facing sensor as shown in this sequence of three images.

real time.

Finally, autonomous robot systems often create visualizations for monitoring the robot or controlling it when necessary. The Virtual Environment Vehicle Interface (VEVI) system is a good example of 3D visualization for robotic applications [3].

Our approach differs from this related work in significant ways. First, we address the real-time and online needs of tele-operation of vehicles at high speeds. Second, we focus on a complete model of the environment, including ground and non-ground objects, and near- and far-field regions. Finally, we are interested in photo-realistic visualization rather than geometrically accurate modeling, which changes the emphasis of the modeling approach.

10. Summary and conclusions

We have described a method for creating photo-realistic models of real-world environments using the fusion of 3D data from laser scanners and 2D imagery from cameras. The method combines several techniques, including estimation and modeling of the ground surface, segmentation of non-ground points from ground points, modeling non-ground points using point interpolation or sets of textured occupied voxels, and modeling of distant surfaces using billboards. Together, these methods allow real-time 3D modeling of the environment surrounding a mobile platform for the purpose of improving tele-operation capabilities.

Our approach offers many advantages over traditional methods of tele-operating vehicles, including the ability to record and visualize information that lies outside the current sensor field of view, the ability to view the scene from viewpoints that are different from the original camera viewpoint, and the ability to modularize and separate the processes of data transmission, world modeling, and visualization.

The modeling process can be improved in many ways, and these are the subject of future research. First and foremost, we have implicitly assumed that the scene is static. Moving objects add an extra level of complexity, since they must be segmented and tracked individually. However, this problem has been studied by other researchers, so we are confident that our method can be extended to handle moving objects. Second, the 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 investigating ways to fuse stereo and laser data for this purpose.

References

[1] S. F. El-Hakim, P. Boulanger, F. Blais, and J.-A. Beraldin. A system for indoor 3D mapping and virtual environments. In *Proceedings of Videometrics V (SPIE v. 3174)*, pages 21–35, July 1997. 7

[2] C. Früh, S. Jain, and A. Zakhor. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *International Journal of Computer Vision (IJCV)*, 61(2):159–184, 2005. 1, 7

[3] B. Hine, P. Hontalas, T. W. Fong, L. Piguet, E. Nygren, and A. Kline. VEVI: A virtual environment teleoperations interface for planetary exploration. In *SAE 25th International Conference on Environmental Systems*, July 1995. 8

[4] N. Ho and R. Jarvis. Large scale 3D environmental modelling for stereoscopic walk-through visualisation. In *3DTV07*, pages 1–4, 2007. 1, 7

[5] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 1, pages 654–661, Oct. 2005. 7

[6] S. Y. Jinhui Hu and U. Neumann. Approaches to large-scale urban modeling. *IEEE Computer Graphics and Applications (CG&A)*, 23(6):62–69, Nov. 2003. 1, 7

[7] M. B. Josh Johnston, Joel Alberts and J. Edwards. Manipulator autonomy for eod robots. In *Proceedings of the 26th Army Science Conference*, Dec. 2008. 2, 7

[8] T. Kanade, P. Rander, and P. J. Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(1):34–47, Jan. 1997. 7

[9] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Proceedings of ACM SIGGRAPH*, pages 39–46, 1995. 7

[10] H. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical Report CMU-RI-TR-96-34, Robotics Institute, Pittsburgh, PA, September 1996. 5

[11] P. Rander. *A Multi-Camera Method for 3D Digitization of Dynamic, Real-World Events*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 1998. 4

[12] M. Schneider and R. Klein. Enhancing textured digital elevation models using photographs. In *Intl. Symp. on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 261–268, June 2008. 7

[13] M. Segal, C. Korobkin, R. van Widenfelt, J. Foran, and P. Haeberli. Fast shadows and lighting effects using texture mapping. *Proceedings of ACM SIGGRAPH*, 26(2):249–252, 1992. 6

[14] I. Stamos and P. K. Allen. Geometry and texture recovery of scenes of large scale. *Computer Vision and Image Understanding (CVIU)*, 88(2):94–118, Nov. 2002. 7

[15] C. Wellington, A. Courville, and T. Stentz. A generative model of terrain for autonomous navigation in vegetation. *Intl. Journal of Robotics Research (IJRR)*, 25(1):1287–1304, December 2006. 4

[16] L. Williams. Casting curved shadows on curved surfaces. *Proceedings of ACM SIGGRAPH*, 12(3):270–274, 1978. 5