Wide-Area 3D Tracking from Omnivision Video and 3D Structure

Olivier Koch KOCH@CSAIL.MIT.EDU
Seth Teller TELLER@CSAIL.MIT.EDU
MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge MA, 02139 USA

1. Introduction

We present a work-in-progress method for real-time vision-based localization in indoor environments. Given a rough model of the structure and a video sequence captured from an omnivision camera, the system computes the camera pose (translation and rotation) in the structure coordinate frame. The system has several novel aspects: it performs localization in 3D; it handles highly cluttered and dynamic environments; it scales well over wide-space areas made of several buildings and it runs over several hours without breaking. We demonstrate that the localization problem can be split into two distinct problems: an initialization phase and a maintenance phase. In the initialization phase, the system determines the camera pose with no other information than the estimate provided by the user (building, floor, area, room). In the maintenance phase, the system keeps track of the camera pose from frame to frame without any user interaction.

2. Related Work

Map-aided localization using vision has been an active field of research over the past few decades. Seminal works by Horn (Horn, 1986), Faugeras (Faugeras et al., 2001) and later Hartley and Zisserman (Hartley & Zisserman, 2004) describe the theoretical aspects of the problem. Since then, many various approaches have been explored and have brought promising results (Ansar & Daniilidis, 2002), (Drummond & Cipolla, 2002),(Bartoli & Sturm, 2005), (Rosten & Drummond, 2005), (Nister et al., 2004). Nevertheless, these methods are often restricted in space and are not very well suited for real world applications.

3. Localization from Correspondences

Our system works by maintaining a set of correspondences between the 3D model lines and the 2D observed edges.

In the initialization phase, the system initializes a set of correspondences from the void and computes an initial camera pose. The model is divided into a set of regularly-spaced nodes. At each node position, the system determines the best camera rotation possible and computes a score function Ψ. At the end of the process, the position with the highest score is retained as the most probable true camera position.

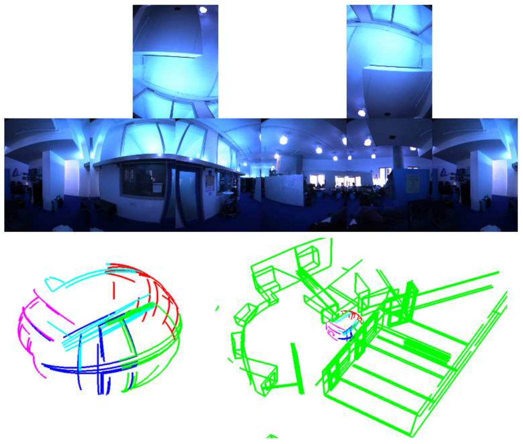
Figure 1. An omnivision image taken in the MIT Stata Center. Raw image (top), detected edges (lower left) and 3D localization (lower right).

The algorithm works by taking every pair of 3D model lines and every pair of 2D observed edges and compute the minimal rotation that would bring the edges in alignment with the lines. At the end of the process, a large list of candidate rotations is obtained, from which the system extracts the ”best rotation”, namely the rotation having the highest density of votes.

The scoring function Ψ measures how well a set of model lines matches a set of observed edges. The function is simply defined as the sum of the best possible score for each model line, where the score is a function of the bearing error between the model line and the observed edge.

In the maintenance phase, the system needs only update the correspondences to compute the new camera pose. For each model line, a set of candidate image line matches is computed. Only one (or none) of these candidates is the correct one. To filter out mismatches, the system tries several configurations, computes the camera pose based on the corresponding assumption and keep the best one in the sense of Ψ. Additionally, the edge tracker may keep track of color information to minimize the risk of mismatches between two consecutive frames.

4. Visibility Precomputation

One major features of our system is the off-line computation of the set of visible faces, edges and vertices at various locations of the model. This allows to bound the search space to a very reasonable size during the initialization and the maintenance phase.

We propose an algorithm based on openGL to take advantage of the attractive computation power of modern GPUs. At each selected location of the model, the system renders the complete model assigning a different color to each face, then parses the image and declares as visible any face which color appears on the image. This algorithm is very reliable and much faster than any geometric technique we could develop. For instance, the complete MIT campus can be covered in about 24 hours with a node every 50 inches.

As for model lines, we propose an algorithm based on the feedback buffer of openGL. The feedback buffer is a special buffer in which openGL stores objects to be displayed along with their depth value, without actually displaying them on the screen. For each line, the depth value of the feedback buffer is compared with the Z-value at various points along the line. If a certain amount of them, say 20%, match, then the line is declared as visible. Figure 2 illustrates our algorithm.

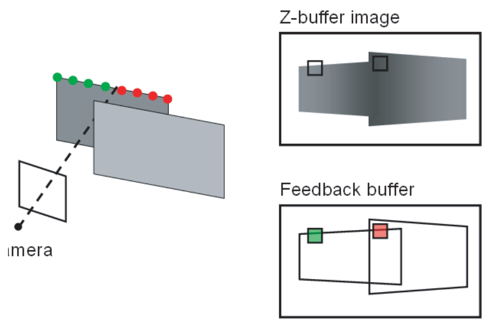

Figure 2. Using the GPU for visibility computation: visible points are shown in green, occluded points in red.

5. Results and Conclusion

Figure 3 shows the 3D localization of the camera across an office space. The flickering observed in the camera motion is due to wrong correspondences during feature matching and to inaccuracies in the 3D model. Synthetic experiments simulating noise and clutter show that the camera position can be tracked up to a few inches in translation and less than a degree in rotation. We are currently working on various techniques to improve the robustness of the feature matching algorithm.

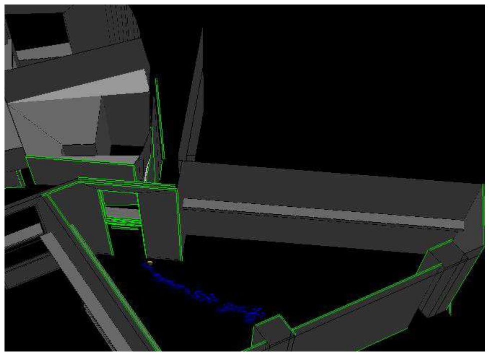

Figure 3. Reconstruction of the camera motion in 3D. The path of the camera is shown in blue. Visible lines used for feature matching are shown in green.

References

Ansar, A., & Daniilidis, K. (2002). Linear pose estimation from points or lines. *ECCV, edited by A. Heyden et al., Copenhagen, Denmark*, *4*, 282–296.

Bartoli, A., & Sturm, P. (2005). Structure from motion using lines: Representation, triangulation and bundle adjustment. *Computer Vision and Image Understanding*, *100*, 416–441.

Drummond, T., & Cipolla, R. (2002). Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, *24*, 932–946.

Faugeras, O., Luong, Q.-T., & Papadopoulou, T. (2001). *The geometry of multiple images: The laws that govern the formation of images of a scene and some of their applications*. Cambridge, MA, USA: MIT Press.

Hartley, R. I., & Zisserman, A. (2004). *Multiple view geometry in computer vision*. Cambridge University Press, ISBN: 0521540518. Second edition.

Horn, B. K. P. (1986). *Robot vision*. Cambridge, MA, USA: MIT Press.

Nister, D., Naroditsky, O., & Bergen, J. (2004). Visual odometry. *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)* (pp. 652–659).

Rosten, E., & Drummond, T. (2005). Fusing points and lines for high performance tracking. *IEEE International Conference on Computer Vision*, *2*, 1508–1515.