Practical 3D Tracking Using Low-Cost Cameras

Roman Barták, Michal Koutný, David Obdržálek
Charles University in Prague, Czech Republic
bartak@ktiml.mff.cuni.cz (corresponding)

Abstract

There exist solutions for tracking of objects in 3D space involving hi-tech cameras and powerful computer systems capable of tracking many objects in large dynamic space simultaneously in real time. On the other hand, there are situations where such functionality is not necessary and the conditions may be specified in more detail, which makes the task significantly easier. This paper shows the possibility to track a single object using low-cost cameras on an ordinary laptop in a small-scale and mostly static environment. This solution is useful for standalone tracking in mobile robotics and particularly in the debugging phases, where the user needs to judge the robot movement system independently on what the robot claims.

1 Introduction

The cost of robotic hardware is going down and inexpensive robotic platforms are now accessible to many researchers. This increases the number of teams doing research in the area of mobile robotics, which raises the needs also for inexpensive supporting tools. For example, research in mobile robotics requires some system for tracking objects in 3D space to have a "ground truth" to which robot navigation and localization systems can be compared.

There exist very precise commercial systems for 3D object tracking. The most notable two are the Hawk-Eye [Hawkins, 2016] and the Vicon [2016] systems. However, due to their price tag they are not accessible to everyone. On the low-cost side, several projects such as the Microsoft Kinect [2016] and LeapMotion [2016] sensors can provide depth information, but for acquiring tracked object location, further processing would be necessary. Also, both these sensors use active infrared projection or illumination, which is very problematic in applications where ambient light may contain this part of spectrum. For example, scenes with direct sunshine are very hard or even impossible to be observed by these two systems.

Research is supported by the Czech Science Foundation under the project P103-15-19877S.

In this paper we describe a low-cost system called Dove-Eye for tracking an object in 3D space. The primary task was to design a passive system that observes the scene using low-cost cameras such as web cameras, uses an ordinary (Linux) laptop for data processing and visualization, and that is easy-to-use without complicated setting. We focus on tracking of a single object that is easy to recognize (such as a green ball in an environment with little or none green color) and that is moving slowly in a given scene observed by static cameras. The ambition is showing how existing computer vision techniques can be integrated to solve the above task – to get a system applicable in real-life environment without the hassle of complicated setting and calibration. From the theoretical point of view, most of the algorithms used are already well known. We present a practical way of assembling all the pieces to create a system working in real world and usable out of the box.

2 System Usage and Setting

The presented system Dove-Eye [Koutný, 2016] works as follows. Let us use a set of fixed inexpensive cameras observing the scene from different points. The smallest setup consists of two cameras (to get stereovision) connected to USB ports; more than four or five cameras is not recommended due to requiring more computational power without bringing significantly improved results. The cameras are calibrated fully automatically by capturing a known pattern that is used to deduce relative location and distances between the cameras (Figure 1). After the calibration, a specific object to be tracked is manually selected in the video frame of each camera – the object is supposed to be easy to recognize by existing computer vision techniques, for example by using a specific color that does not appear elsewhere in the scene. The lines pointing towards the object from each camera give the location of the selected object in 3D (in theory, in the intersection of these lines; in practice the lines may not intersect so the point nearest to these lines is used instead). The calculated position is added to the so far constructed object location set. Consecutively, as the object moves, the trajectory is drawn in the viewer using this set, forming a virtual 3D path of the tracked object (Figure 2).

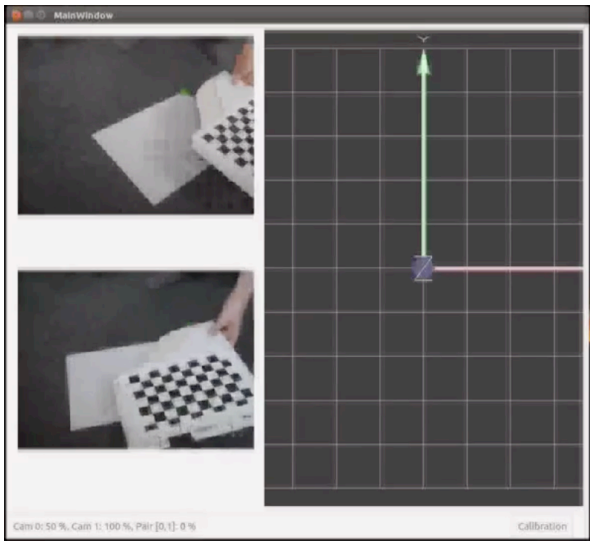

Figure 1: Calibration of cameras and their pairs

3 Realization and Implementation

The application uses the OpenCV computer vision library [Itseez, 2016] and connects everything together, bringing the user-friendly solution for setting up the system, tracking, and logging the object movement. At the application start, the user selects the cameras, which are to be used as image input sources for the tracking. From now on, the main window will show individual images (streams) from all selected cameras plus a 3D viewer of the scene where the 3D tracking output will be shown (Figures 1 and 2).

3.1 Calibration

The application firstly calibrates the setup by calibrating individually all cameras one by one and then calibrating relative positions of each possible pair of cameras. For the pattern, we have selected the chessboard pattern. During the calibration, this pattern is looked for on the image using *findChessboardCorners* and if a match is found, the position of this pattern is passed to the *calibrateCamera* and *stereoCalibrate* functions. Single camera calibration gives for each camera its matrix of intrinsic parameters, radial distortion, and tangential distortion. The user simply puts the printed pattern in the field of vision of every camera and the system automatically calibrates individual cameras and notifies the user about the progress in the application status line (Figure 1). Stereo calibration calculates the relative position of each pair of cameras (i.e. the transformation between their coordinate systems as rotation and translation matrices) and the fundamental matrix depicting the relation between corresponding points in the two camera outputs. The calibration pattern must be positioned in the scene so that it is well visible on a specific pair of cameras. Analogous to the single camera calibration, the system performs the calibration automatically and notifies the user in the application status line.

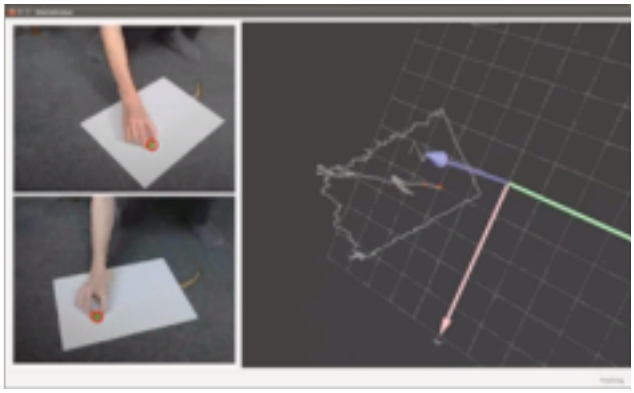

Figure 2: Tracking: The user drags the object on the floor.

3.2 Tracking

For the tracking, the user has to select the object to be tracked. This is done by marking a rectangular area on each of the camera outputs (Figure 2). Color-based tracker which uses color histogram of the originally marked object and backprojection on the consecutive video frames proved to work reliably provided that objects featuring similar colors were not present in the scene background. Kalman filter is used for movement estimation. For the base movement prediction, linear approximation was used as a first fast stand-by. When the object is tracked on individual camera images, 3D localization is performed. This is done by exploiting the *triangulatePoints* OpenCV function for each pair of cameras. For more cameras, a simple centroid of individually calculated positions is used.

4 Summary

The presented project brings an easy-to-use tool for visual tracking of a selected object in 3D space. The tool can be exploited without complicated setting in any environment and it is particularly dedicated to mobile robotics researchers to provide independent tracking of robot's movement.

References

[Hawkins, 2016] Paul Hawkins, Hawk-Eye Innovations (Sony group), *Hawk-Eye vision processing system,* http://www.hawkeyeinnovations.co.uk

[Itseez, 2016] Itseez. *OpenCV Open Source Computer Vision Library*, http://www.opencv.org

[Kinect, 2016] *Microsoft Kinect Sensor SDK*, https://dev.windows.com/en-us/kinect

[Koutný, 2016] Michal Koutný, *Dove-Eye system,* http://koutny.org/dove-eye/

[Leapmotion, 2016] Leap Motion, Inc., *Leap Motion Controller*, https://www.leapmotion.com

[Vicon, 2016] Vicon Motion Systems Ltd., *Vicon Motion Capture System,* http://www.vicon.com