Application of augmented reality for supporting instrument service tasks

Toni Benussi†  Zoran Kalafatic‡  Zeljka Mihajlovic‡
University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia

Abstract

This paper describes a system for object detection and tracking implemented as an iPad application, which has been tested and successfully accomplishes its task. The system is based on finding the initial position of the object by matching features between a template and the device’s picture. Tracking is achieved by using a pyramidal implementation of the iterative Lucas-Kanade algorithm. The implemented system was tested on two different instruments used for testing internal combustion engines. The paper discusses the possibilities offered by mobile devices, like the iPad, for the development of applications with computer vision and augmented reality elements and also describes the major problems that have been encountered on such platforms.

CR Categories: H.5.1 [Information interfaces and presentation]; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality

Keywords: computer vision, augmented reality, mobile devices, iPad, feature detection, tracking

1 Introduction

In the last few years the processing power of mobile phones and other hand-held devices, like tablets, have increased dramatically which allows more and more complex tasks to be achieved using such devices. One of the new possibilities is the use of mobile devices for real time processing of images and video sequences. The motivation for the research described in this paper was the desire to improve the installation and maintenance procedures of automated testbeds used for internal combustion engine testing. In this paper we tried to use a tablet device (iPad3) to detect and track known objects, specifically two peripheral devices of the automated testbed, using already established techniques which have been adapted and carefully parameterized to achieve real-time performances on a tablet. The goal is to overlap the important parts of the tracked object with useful information regarding such parts like technical documentation extracts, maintenance instructions and measured data.

Section 2 shortly reviews the related work on this subject. The implemented algorithm is described in sections 3 and 4, separately the initialization procedure and the tracking procedure. The results are presented in section 6 together with the encountered limitations and challenges present on mobile systems. The conclusion is given in section 7 along with potential future research directions.

2 Related work

An implementation of parallel tracking and mapping on camera phones is described in [Klein and Murray 2009]. The goal in that paper is to create a map of the 3D environment from a series of images based on tracking of natural features. They track FAST [Rosten and Drummond 2006] features while also simultaneously updating the feature map with new features from newly acquired images. Tracking is accomplished by predicting the most likely location of the features based on previous movement speed and direction and the results are refined by local search around the predicted location.

In [Wagner et al. 2008] a feature tracking method is implemented. The method is based on SIFT descriptors proposed in [Lowe 1999], but substitutes the Difference-of-Gaussians (DoG) feature detector with the much faster corner detector from [Rosten and Drummond 2006]. To match SIFT descriptors between the current image and the reference model the authors used Spill Trees described in [Liu et al. 2004]. To calculate the pose of the camera in relation to the tracked object they used Gauss-Newton iterative scheme to minimize the reprojection error to the image plane while taking into account the camera model with radial deformations.

3 Initialization

The goal of the initialization procedure of the algorithm described in this paper is to detect the instrument that will be tracked and determine it’s position and orientation in the first frame from the iPad’s embedded camera. The image acquired from the camera is first transformed into a grayscale image and the same is done for each subsequent frame. The iPad’s camera suffers from high levels of noise, which is typical for embedded cameras. To mitigate this problem the grayscale image is processed with a Gaussian filter. The majority of the basic components and algorithms we used are implemented in the OpenCV library and written in C++, while the GUI was written in Objective C.

The next step is feature detection. Different algorithms were tested (FAST [Rosten and Drummond 2006], SURF [Bay et al. 2008], SIFT [Lowe 1999]), but for the final implementation we used the detector Good features to track (GFTT) [Shi and Tomasi 1994]. The available implementation of the SIFT detector was too slow and was immediately discarded, while the other detectors were tested and the results can be seen in table 1. The SURF detector was too slow for real time application. FAST has great performance and detects a lot of features compared to the other detectors, but

<table>
<thead>
<tr>
<th>Feature Detector</th>
<th>iPad Duration [ms]</th>
<th>PC Duration [ms]</th>
<th>Num. of Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAST</td>
<td>36</td>
<td>7.5</td>
<td>1258</td>
</tr>
<tr>
<td>SURF</td>
<td>3413</td>
<td>670</td>
<td>632</td>
</tr>
<tr>
<td>GFTT</td>
<td>382</td>
<td>83</td>
<td>361</td>
</tr>
</tbody>
</table>
even while using non-maxima suppression the detected features are
tightly grouped together which can cause inaccurate computations
of the homography later on. The GFTT detector has acceptable
computation speed and a lot of parameters to refine the criteria for
feature detection which was the primary reason for our choice.

Feature descriptors are generated using the technique proposed in
[Bay et al. 2008]. The specific objects we were trying to detect
have many similar features on their surfaces, so to distinguish the
features it was necessary to expand the obtained descriptors with
scaled information about the position of the features on the object.

The same process of feature detection and description is performed
offline on a set of template pictures of the object we want to track.
The templates differ from each other by the angle and distance from
which the object is observed. The position of the object on the tem-
plates is known beforehand. The descriptors of the features detected
in the image from the live camera are matched to the features of
each template using the FLANN [Muja and Lowe 2009] algorithm.
The best matching template is chosen and 40 best feature matches
between the image and the template are used to compute the ho-

mography matrix using the RANSAC [Fischler and Bolles 1981]
method. The homography matrix contains the information about
the position of the object on the image in relation to its position on
the template which is enough to start tracking the object and add
virtual objects on top of it.

4 Tracking

For tracking we used an iterative implementation of the Lucas-
Kanade algorithm [Bouguet 2001] that uses a pyramidal represen-
tation. We track 20 most prominent features obtained with [Shi and
Tomasi 1994]. Note that these features are not necessarily a subset
of the 40 features used in the initialization as they are not chosen
by the criteria of similarity with a template, but exclusively because
they are very suitable for tracking.

The positions of the features on the current image and their coun-
terparts on the previous image are used to compute the homography
matrix between the two images. This solution to compute the ho-
mography between two close images has proven to be much more
precise in comparison to computing the homography between the
current image in the video sequence and the initial template, as it’s
easier to compute the homography for smaller changes of the view-
point. Sometimes during the tracking procedure features are lost
between frames. Those features are excluded from the set used to
compute the homography. To restore the lost features we use the
homography computed from the successfully tracked features. As-
suming that the homography is correct the lost features are restored
to their actual position in the current image. This procedure allows
us to maintain a constant number of features and to track the object
for longer sessions.

5 Virtual objects

By successfully tracking the object’s position in the scene we are
given the means to overlap parts of the objects with virtual objects
containing useful information about them and create an augmented
reality environment. The idea is to allow the user to interact with the
components of the tracked device, in this case the MicroSoot smoke
meter and the F-FEM-CON signal processor. The virtual elements
allow the user to instantly get useful extracts from the manual or
technical documentation specific to each component of the device
g or get real-time measurements and readings on the device from the
central workstation which is connected to the iPad by a wireless
network.

By knowing the position of the tracked device in the current image
we can determine the exact position of each of the device’s compo-
nents and add the required information to the appropriate position
in the image. Examples can be seen in figure 1.

![MicroSoot with measurement values](image1a)

![F-FEM-CON with manual extract](image1b)

Figure 1: Images of the devices with augmented reality elements

6 Results

"The two algorithm components, initialization and tracking, were
separately tested. The initialization was tested on 90 pictures for
each device and the result are shown in table 2. The results for
the F-FEM-CON device are significantly worse then the Microsoot
results because the F-FEM-CON has a large number of very simi-
lar features which causes mistakes during the matching procedure
which yields a poorer homography. Testing of the tracking proce-

Table 2: Initialization evaluation

<table>
<thead>
<tr>
<th>Device</th>
<th>Successful</th>
<th>Unsuccessful</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroSoot</td>
<td>92.8%</td>
<td>7.2%</td>
</tr>
<tr>
<td>F-FEM-CON</td>
<td>87.75%</td>
<td>12.25%</td>
</tr>
<tr>
<td>Total</td>
<td>90.275%</td>
<td>9.725%</td>
</tr>
</tbody>
</table>

dure was performed on 3 video sequences of the Microsoot device
with the total length of 172 seconds and on 4 video sequences of the
F-FEM-CON device lasting a total of 176 seconds. We recorded
the amount of time the devices were successfully tracked and also
counted the events of tracker loosing the object and required to be
reinitialized. The main reasons for the loss of the object are the
high levels of motion blur present during camera movements and the narrow field-of-view of the camera which causes the object to leave the scene if larger movements are performed. The tracking evaluation results are presented in table 3. During the tracking procedure the system achieves 9 to 10 frames per second on average.

<table>
<thead>
<tr>
<th>No. device</th>
<th>Tracking lost</th>
<th>Tracking successful</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroSoot</td>
<td>6</td>
<td>80.8%</td>
</tr>
<tr>
<td>F-FEM-CON</td>
<td>5</td>
<td>84.1%</td>
</tr>
<tr>
<td>Total</td>
<td>11</td>
<td>82.5%</td>
</tr>
</tbody>
</table>

7 Conclusions and further work

In this paper we described a system for detecting and tracking objects with a handheld tablet device and added augmented reality components to the tracked object. The implemented system was tested and achieves a success rate of 90.275% in object detection and registration and the tracking procedure was successful in 82.5% of the total frames in the video sequences used for testing. The results are encouraging, however there are many issues still left to solve. The frame rate should be increased by parallelizing parts of the procedures, for example the tracking algorithm and the homography computation could be done in separate threads. Also the possibility to use the GPU of the mobile device for computation purposes could be worth investigating. An interesting solution to reduce the tracking problems caused by motion blur could be to use the embedded inertial sensors which are present in the iPad. Its measurements could be used to predict the camera movement in scenarios where the camera moves too fast for visual tracking to work correctly.

Acknowledgments

We thank the company AVL Graz for acquiring all the necessary equipment for our research and their employees for their ideas and suggestions that contributed to the realization of this paper.

References


