# Automatic Depth Camera Calibration

Semester Project

Mihai Bace
mihai.bace@epfl.ch

*S*upervisor Julien Pilet
julien.pilet@aptarism.com

June 5, 2013

**Abstract**

The purpose of this report is to present a method in which a normal RGB camera can be calibrated using a 3D camera. The 3D camera used for calibration is a Microsoft Kinect which has depth detection capabilities using an infra-red sensor. Using the Kinect, this report will present an approach to calibrate the camera based on image processing techniques.

# Chapter 1

# Introduction

The purpose of this project is to accurately calibrate a non 3D camera using a 3D camera. In this report we try to calibrate a normal camera (e.g.) using a Microsoft Kinect.

## 1.1 About the Kinect

The Kinect is a motion sensing device, initially built as an add-on for the Microsoft XBOX 360 gaming console, but later on ported to the PC. It is a relatively inexpensive device that incorporates a lot of different sensors like an RGB camera, a depth sensor and a multi-array microphone which provide full-body 3D motion capture [Wikipedia]. The depth sensor is a combination of an infrared laser and a monochrome CMOS sensor which can capture 3D data under any lighting conditions.
The sensors being used for this project are:

- RGB camera
- Depth sensor: monochrome CMOS and infrared laser

The RGB camera can capture 8-bit images at a VGA resolution (640 x 480 pixels). However, the hardware is capable of capturing images at higher resolutions (1280 x 1024 pixels), but at a lower frame rate. The frame rate for video capture from the RGB camera can vary between 9 Hz and 30 Hz.
The second sensor used, the depth sensor, is very important for the 3D sensing capabilities of the device. The monochrome camera can capture images at VGA resolution (640 x 480 pixels) with 11-bit depth, which can assure 2,048 levels of sensitivity. The sensor has a practical range between **1.2 meters** and **3.5 meters**. In order to be able to take advantage of the Kinect's motion sensing abilities an area of 6 $m^2$ is needed.
The Kinect connects to the PC or XBOX using a USB interface, but it requires an additional power supply due to the motorized tilt mechanism.
Since its launch, the Kinect has attracted a lot of interest from developers and as a result, Microsoft has decide to release and SDK for Windows in the spring of 2011. The SDK is compatible with Windows 7 and newer versions and it allows developers to build applications with **C++, C# or Visual Basic** in combination with the IDE called Microsoft Visual Studio (version 10 or newer).
Some of the features of the SDK are the following [Wikipedia]:

- Raw sensor streams: Access to low-level streams for the depth sensor, color camera sensor and the microphones

- Skeletal tracking: The capability to track one or two people based on the skeleton and use this information for gesture driven applications (e.g. games, medical applications etc.)
- Audio capabilities: Audio processing capabilities include echo cancellation, acoustic noise suppression and the very interesting speech recognition
- Sample code and documentation: a very good starting point for all the beginners who want to develop applications using the Microsoft Kinect

For those who are interested to develop applications for the Kinect, the latest SDK (at this date) is Microsoft Kinect SDK 1.7 and it can be found at the address below:
`http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx`
Since the Kinect has been received with a lot of enthusiasm by the developers, there are some 3rd party libraries and drivers that can be used. Some of the most famous ones are:

- OpenNI `http://www.openni.org/`
- libfreenect, part of the Open Kinect project `http://openkinect.org/wiki/Main_Page`

## 1.2 Depth camera calibration

The depth camera calibration is a process in which we try to reproject a 3D point to a 2D point captured by the normal RGB camera. Without a calibration this would not be possible. The calibration helps to accurately detect what 2D point from the RGB image corresponds to what 3D point (in our case detected by the Kinect).

## 1.3 Motivation of this project

The process in which a 3D point is reprojected to a 2D image is not new. Depth camera calibration has been done before, but this process implies to manually find a minimum number of correspondences. Two points are valid points for the set of correspondences if the points in the 2D image and the 3D image represent the same physical point (pixel in the image plane). Once the points have been found, we try to estimate the calibration matrix using a model. Based on this model all the other points will be reprojected and thus the calibration is completed.
Finding the correspondences between the 2D image and the 3D image is a tedious work and it implies a lot of time. This is why an automatic depth camera calibration would be useful. By having an automatic way to detect the correspondences, the user only needs to walk with a template in front of the camera and the "good" points will be selected automatically. What exactly are good points? We will describe this in the following chapters.

This document will describe in the following chapters a way in which the automatic depth camera calibration can be achieved. The design chapter describes the process from the algorithm point of view, what methods we have used and why. The implementation chapter will go into more implementation details and will describe the functions that make the algorithm work. In order to be able to asses our results, we need some visualization tools. Chapter 4 will describe the visual interfaces. The 5th chapter will present the final result, the calibration.

# Chapter 2

# Design

# Chapter 3

# Implementation

# Chapter 4

# Visualization

# Chapter 5

# Results