TÉCNICO LISBOA

Forensic use of Mobile Phone Cameras: Measuring the Height of a Person

José António Carvalho Mendes

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Examination Committee

Chairperson: Professor João Fernando Cardoso Silva Sequeira
Supervisor: Professor José António da Cruz Pinto Gaspar
Co-Supervisor: Professor Pedro Daniel dos Santos Miraldo
Member of committee: To be defined later

October 2017

Resumo

Este trabalho aborda a estimativa da altura de uma pessoa numa imagem, capturada por uma câmera RGB num telemóvel. Nesta dissertação, utilizamos o pressuposto que uma câmera auxiliar de profundidade é usada para auxiliar na calibração da câmera do telemóvel que originalmente tirou a fotografia. O metodo de calibração proposto usa o algoritmo Direct Linear Transformation (DLT) com pontos e/ou linhas. O uso de linhas permitirá que os dados de calibração beneficiem dos métodos de processamento de imagem. A estimativa de altura será baseada no modelo de previsão 3D, ou considerando que a pessoa está em pose vertical. A análise de incerteza na estimativa de altura mostra como os parâmetros da câmera (como ângulo de inclinação e zoom) podem tornar a estimativa mais robusta.

Palavras Chave: Calibração de Câmara, DLT-pontos, DLT-linhas, estimação de altura, pose 3D, incerteza.

Abstract

This work addresses the height estimation of a person in a picture, that was captured by a RGB camera on a mobile phone. In this dissertation, we use the assumption that an auxiliary mobile color-depth camera is used to aid in the calibration of the phone camera that originally took the picture. The proposed calibration method use the Direct Linear Transformation (DLT) algorithm with points and/or lines. The use of lines will allow the calibration data to benefit from image processing fitting and detection tools, improving the results. The height estimation will be based on 3D pose prediction model, or considering that the person is in vertical pose. Uncertainty analysis in height estimation shows how camera parameters (such as tilting angle and zoom) can make the estimate more robust.

Keywords: Camera calibration, DLT-points, DLT-lines, height estimation, 3D body pose, uncertainty.

Contents

Resumo **i**

Abstract **iii**

1 Introduction **1**

1.1 Related Work 2

1.2 Problem Formulation 4

1.3 Thesis Structure and Contributions 5

2 Background **7**

2.1 Camera Projection Model and Back-projection 7

2.1.1 Projection Model 8

2.1.2 Back Projection 9

2.2 DLT Based Camera Calibration 10

2.2.1 DLT Points 10

2.2.2 DLT Lines 12

2.3 Human Pose Estimation 14

2.3.1 Image based 2D pose estimate 14

2.3.2 3D model matching 16

3 Camera Calibration **17**

3.1 Color-Depth Auxiliary Camera 17

3.2 Calibration Methods using Color and Depth Data 18

3.2.1 Set 3D Data to a Ground Plane Reference Frame 19

3.2.2 DLT-points based Calibration 19

3.2.3 DLT-lines based Calibration 19

3.3 Assisted Matching of Lines 20

3.4 Automated Calibration 21

4 Height Measurement Methodologies **25**
4.1 Measuring the Height of a Person 25
4.1.1 Ground point between feet and optic ray tangent to the head 25
4.1.2 Height estimation 26
4.1.3 Height Estimation Method Summary 27
4.2 Non-Vertical Body Pose 27
4.2.1 Up-to Scale Pose 29
4.2.2 Metric 3D Pose 29
4.2.3 Summary of Height Estimation for Non-Vertical Poses 31

5 Experiments and Results **33**
5.1 Calibration Results 33
5.2 Non-Vertical Height Measurement 35
5.3 Height Measurement Uncertainty Study 36
5.3.1 Noisy data SIFT and noisy SIFT based calibration 37
5.3.2 Height estimation vs depth 37
5.3.3 Height estimation vs tilting 38
5.4 Real-World Datasets 38
5.4.1 Measures with Automated Calibration 39
5.4.2 Person height measure with Assisted Matching of Lines 39

6 Conclusion and Future Work **45**

A Estimate Angle between two Rotation Matrices **47**
A.1 Rotation matrix properties 47
A.1.1 The columns of a rotation matrix are orthogonal unit vectors 47
A.1.2 The transpose of a rotation matrix is its inverse 48
A.1.3 Geometric aspects of the exponential and logarithm 48
A.2 Rotation between two matrices 49

B Projection Matrix Decomposition **51**
B.1 QR based on Gram-Schmidt orthonormalization 51
B.2 Converting QR to RQ 52
B.3 Correcting the diagonal K 52

List of Figures

1.1 Representation of the problem addressed in this thesis. Person image captured from uncalibrated camera at left, followed by calibration of such camera using an auxiliary Kinect camera. At right, it is represented the height estimation using just calibrated camera. 4

2.1 Representation of the pin-hole camera model with respective camera frame (u, v), world frame (X, Y, Z) and projection of 2D point in image plane. . . . 8

2.2 The classic articulated limb model of Marr and Nishihara, [26] . In the middle, the different orientation and foreshortening states of a limb, each of which is evaluated separately in classic articulated body models. On the right, these transformations with a mixture of non-oriented pictorial structures, in this case tuned to represent near-vertical and near-horizontal limbs, from [31]. 14

2.3 Average HOG feature, from [31], as a polar histogram over 18 gradient orientation channels as computed from the PASCAL 2010 dataset. On average, images contain more horizontal gradients than vertical gradients, and much stronger horizontal gradients as compared to diagonal gradients. 14

3.1 Bouguet calibration setup. A checkerboard is imaged at various poses. 18

3.2 Calibration helped by a color-depth (RGBD) camera. 18

3.3 DLT-Points setup. Points in 3D are matched with image points in 2D to obtain camera calibration. 19

3.4 DLT-Lines setup. Fitted Lines in 3D are matched with tuned image lines in 2D to obtain camera calibration. 20

3.5 Methodology for picking Lines helping in user input. Best line configuration is found by minimizing the cost function $C = -\sum_k \|\nabla I(m_k)\|$. 20

3.6 Assisted Matching of Lines. Methodology where user input is required to perform matching between 3D and 2D lines detected. DLT-Lines is then used to perform camera calibration estimate. 22
3.7 3D line fitting using RANSAC. 23
3.8 Automated Calibration. Camera Calibration methodology using SIFT to obtain a 1st estimate which will help in line matching between images in order to perform calibration. 24

4.1 Back-projection line of user clicked on point on floor with respective projection on vertical, L1. While L2 is the back-projection line of user clicked point in head. Closest point in each one of them is respectively c1 an c2. 26
4.2 3D pose model estimation from scaled normalized body $\widetilde{M}$. C stands for aligned camera frame while W stands for world frame where real camera projection, P_w is valid. 28
4.3 3D pose model estimation using aligned camera frame, C. Z_{DS} stands for 3rd coordinate in camera frame in normalized body, while Z_{cam} stands for 3rd coordinate in real camera, which means distance from camera center to person foot. 31
4.4 3D pose model estimated in (a) using (2.27) for a scaled z, while in (b) it is used (4.15) providing metric units, making it possible to estimate height by adding all magenta segments. 32

5.1 Camera calibration setup used in VRML (a), RGB image in (b), depth-color image in (c), and respective depth map in (d). 34
5.2 SIFT results obtained between RGB and depth-color image. 35
5.3 Lines re-projected and fitted using first projection matrix estimate. In red are simple reprojected lines while in blue are lines fitted with image gradient. . . . 35
5.4 Calibration results obtained, RGBD camera in red, ground truth in blue, 1st estimate in green, 2nd improved estimate with lines in magenta. 36
5.5 2D person in vertical pose for comparison height estimation in (a), followed by person in non-vertical position detection in (b), that will lead to joints location estimation in (c) 36
5.6 3D Pose estimate with aligned camera in black and camera who took the picture in red. 37
5.7 Different depth of character in scenario in (a), person detection results in (b) for different depths, 3D model predicted (c) from 2D detection in one of the cases. 38

5.8 Uniformly distributed noise applied to user clicked (user point in orange, noisy points in green) in (b) and noise applied to user click and calibration data in (c). 40
5.13 Height Estimation of person standing using RGB image obtained with mobile phone camera and calibration data from Assited Matching Lines method (a).3D predicted model from person in image in (b).Results showing estimated mobile phone camera position over the pointcloud (c). 40
5.9 Different depth in hall starting at 3m from origin frame on the left wall going up to 6m (a), with respective height mean and std dev in (c), different depth in corridor starting at 7m and going to 12.5m (b), with respective height mean and std dev in (d). 41
5.10 Different tilting as it can be seen in (a) from 0° to 30°, respective estimated heights in (b) and their uncertainty in (c). 42
5.11 Calibration of a fixed surveillance camera using a mobile color-depth (RGBD) camera. (a) RGB image of calibrated camera to calibrate. (b) and (c) show color and depth images acquired by a Microsoft Kinect. (d) Results showing the location of the color-depth camera (red) and estimates of the locations of surveillance camera, 1st estimate in green and 2nd estimate in magenta, all drawn over the point cloud acquired by the color-depth camera. 43
5.12 Clicked points by user in blue and orange to perform vertical height estimate. On the right value of air conditioning height. 44
5.14 Height Estimation of person sitting using RGB image obtained with mobile phone camera and calibration data from Assited Matching Lines method (a).Depth-color image obtained with Microsoft Kinectic (b).Results showing estimated mobile phone camera position over the pointcloud (c). 2D person members detection in (d). 3D predicted model in (e) from results in (d). 3D model predicted with 2D joints positions aided by the user(f). 44

List of Tables

5.1 Height Estimate results VRML 37
5.2 Height Estimate results in Unity 39
5.3 Height Estimate Results Real Data 42

Chapter 1

Introduction

The increasing need of surveillance in public spaces, and the recent technological advances on embedded video compression & communications, made camera networks ubiquitous. Nowadays technological advances already allowed such a wide installation of camera networks. However the calibration of these cameras, considering an unique reference frame, is still an active research topic. These calibration parameters are essential for further higher level processing, such as: people/car tracking, event detection, and metrology, i.e. some of the most active research subjects in Computer Vision / Video Surveillance.

Many times, surveillance and mobile phone cameras capture suspects that need to be arrested. A key step for re-identifying those suspects is the image based measurement of biometric data, such as the body height. Even though the quantity of cameras is increasing, in many cases, the quality of the video footage is still too poor to identify these suspects (the quality of these images is not clear enough for facial recognition). Methods developed so far use simple ratios to make approximate estimate of heights. Even when the suspect try to minimize his biometric signature, the perpetrator's vertical height is calculated in the image to make an estimation of his actual body height (no consideration is taken to the subject's posture). Due to different postures, individual variations in standing & gait, and loss of information when the 3D reality is captured in a 2D picture, the body height estimation from surveillance camera images is difficult.

The technological advances already allowed such a ubiquitous presence of cameras. However, the still infancy stage of video analytic and the high variability of forensic research challenges make, *per si*, the extraction of information from the cameras an active research area. The work described within this dissertation aims at developing methodologies for measuring heights of people, given video streams. Typical environments include single rooms, complete

buildings, streets, highways, tunnels, etc. For these type of scenarios, one of the crucial problems is to estimate the camera pose in the scenario, that captured the image of the suspect. After the computation of the camera's location on the scenario, the height of the suspect can be estimated (assuming some errors due to the person's posture). Given the forensic nature of the application, it is assumed that the geometric of the scene data can be obtained after the video acquisition.

1.1 Related Work

Conventional calibration methodologies (such as the one proposed by Tsai [29], Heikkila [17], Zhang [32], or Bouguet [4]) require a known pattern in image. Precise calibration demands that the know pattern is covering most of the imaging area, meaning that it will be impractically large if camera is mounted at high position. Also the methodologies used for camera calibration are mostly focused in the intrinsic parameters, and thus do not provide distance (rigid pose transformations representing the extrinsic parameters) among various cameras. They are not designed to provide a global coordinate system for all cameras.

Creating a global coordinate system for a set of cameras having a non-overlapping fields of view (FOV) has been approached in works such as [23] and [20]. In [23], a mobile robot, with the capability of estimating its pose in a global frame, is responsible for transforming its coordinate system to a set of fixed cameras. This mobile platform, equipped with a calibrated camera, estimates its position and orientation using visual Simultaneous Localization And Mapping (vSLAM). Correspondences between Scale Invariant Features (SIFT), detected on the mobile and fixed camera images, allow the estimation of the intrinsic and extrinsic parameters of the cameras in the network. In [20], information from two cameras with non-overlapping fields of view, supplies up-to-scale 6D motion information, while the metric scale is recovered via a linear solution, by imposing the known static transformation between both sensors. The redundancy in the motion estimates is finally exploited by a statistical fusion to an optimal 6D metric result. In [22], it is presented an efficient computational technique in order to estimate relative pose for multi-camera systems. The problem is faced as a low-dimensional iterative optimization, over the relative rotation only, directly derived from the well known epipolar constraint. Generalized camera models, [12] are also used to model camera clusters.

Laser Range Finders (LRF), combined with SLAM, proved to provide reliable scene information (3D clouds of points) of large areas [18]. These 3D maps can, therefore, be used to provide required data to calibrate a camera, by selecting a region of interest on the map.

Recently, color-depth cameras (also known as RGBD cameras) have become an interesting low cost alternative to LRF sensors, [9]. A set of 3D points is simply acquired by back-projecting 2D points from the RGBD image plane. Features can be detected in a network camera image and, then, matched with the RGBD image points. This defines a set of 2D-to-3D points correspondences, which can be used to estimate the camera projection matrix using the Direct Linear Transformation (DLT) [2, 13].

The use of 2D lines' images allows some methods for line fitting to be added to the DLT. The linear constraints for calibration from line correspondences (used to estimate the projection matrix) are as simple as with points [28]. Also, despite the low computational complexity of the proposed calibration, the results based on lines provided some similar accuracy. The ability to assess the uncertainty in the estimates of a calibration method is important, not only to infer errors on 3D reconstruction but also as a means to validate and improve the calibration process. This level of accuracy is analyzed in [11], when calibration based on DLT-Lines is used, showing how the uncertainty propagates from the measurement process to the uncertainty on the calibration parameters (either intrinsic, extrinsic, or 3D reconstruction). Validation was done with synthetic data, based on Monte-Carlo simulations, and with real data, from a non-overlapped camera network.

In [30], it is presented an effective calibration method based on planar template methodology. The described method is based on the two step method of Tsai's,[29], improving it computationally efficient by using co-planar points. They show that camera calibration is a very important part of the non-contact human body measurement system.

About height estimation in body measurement, in [21], it is shown how it is possible to perform height estimation and, in order to have more sophisticated surveillance videos, the importance to track people. Height has been long used in forensic procedures to narrow the list of possible suspects (it is not distinctive enough to be used in biometric identification). However, by estimating the heights of tracked subjects on different cameras, it could provide an important additional feature, allowing a more robust estimation of the tracking object over different scenes. This work introduces a method to estimate the height of human subjects tracked on calibrated surveillance camera images, using estimated parameters such as camera altitude & tilting, and knowing the top & bottom farthest points from person in an image.

In the topic of estimating the person's pose, using static images, there are some works in the literature. For example, in [1], the detection and articulated pose estimation uses a trained appearance model, and a flexible kinematic tree prior of body configurations parts, to estimate 2D pose. Some extension of tree-based models were proposed, using part mixtures for capturing contextual (co-occurrence relation between parts) and spacial relations with histogram of gra-

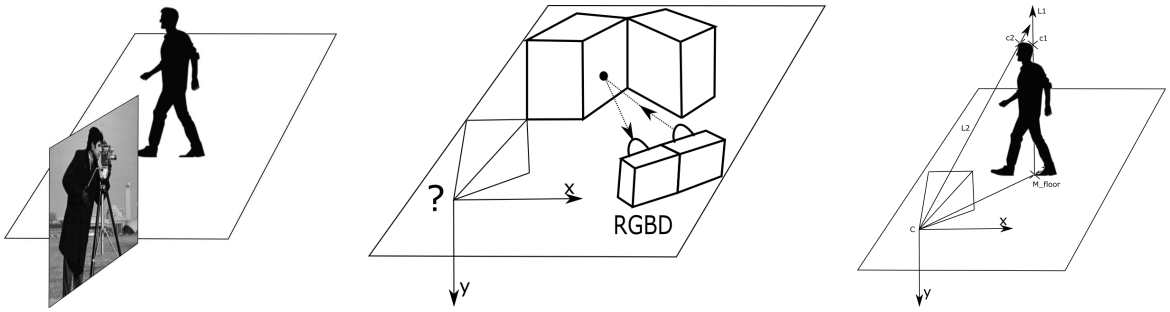


Figure 1.1: Representation of the problem addressed in this thesis. Person image captured from uncalibrated camera at left, followed by calibration of such camera using an auxiliary Kinect camera. At right, it is represented the height estimation using just calibrated camera.

dients ([26]) to estimate 2D pose, [31]. The computational advances of Deep Neural Networks have revolutionized 2D pose estimation, producing accurate 2D predictions, even for poses with self-occlusions [6]. With the "Big-data" sets of 3D information, it is possible to predict the 3D pose from the 2D pose, using simple memorization [6] (nearest neighbor model).

1.2 Problem Formulation

As shown in [21], it is possible to perform height estimation from a calibrated camera, with the assumption that some parameters are known. Since a minimum precise calibration can be performed using methods [28], with data acquired from LRF or from color-depth camera, [18, 9], such calibration can then be used to perform an height estimate, see Fig. 1.1. In this thesis it is uses the SIFT algorithm, as in [23], in order to perform a camera calibration estimate that will use DLT-Lines. An uncertainty analysis in the estimates of a height is made. This is important not only to use it as biometric data (to identify possible suspect) but also as a means to control the pan and tilt angle of surveillance camera, in order to minimize the uncertainty in measures.

In [21], height estimate is made considering that the person is standing. In this thesis, it is used a tree-based model of part mixtures for capturing contextual co-occurrence relation between parts and spacial relations, with histogram of gradients to estimate 2D pose, [31]. Having a 2D pose approximation, it is used a "nearest neighbor" algorithm to perform matching with library from [6], in order to estimate 3D pose. This will allow the estimation of the person height, in cases where non-vertical poses are considered.

1.3 Thesis Structure and Contributions

In this section I will present the thesis structure and its contributions. Chapter 2 gives a description on the background theory in camera calibration topics, such as projection, back-projection, and DLT calibration techniques based on points and lines. Chapter 3 introduces the proposed methods and setups used for camera calibration. Chapter 4 presents the techniques used to obtain forensic data in an image of a calibrated camera (in this case height of person). Chapter 5 shows some experiments and results on real and simulated setups and Chapter 6 presents the conclusions and future work.

Appendix A describe the algorithm used to estimate angle between two rotation matrices, and Appendix B shows how to decompose projection matrix.

The main contributions of this thesis are: (i) Automation of process of calibrating color camera with the help of depth camera; (ii) Estimating the height of a person given the prior camera calibrate methodologies and public domain libraries to find person and there up to scale skeleton pose.

Chapter 2

Background

This section presents the most important background techniques that were used in the development of this work.

Considering the aspects of camera calibration, we follow, in essence, the notation and methodologies from M. Silva [27, 28]. First, the projection model will be introduced and described, such as the back projection technique, i.e. the relation between image points and 3D world points [14]. Then, some methods for camera calibration will be introduced, such as the DLT Points and the DLT Lines calibration methods. While some methods use the camera motion and the static scene structure to perform the respective camera calibration, e.g. [23, 5, 16], others combine 3D and 2D information to estimate camera parameters, e.g. [13, 2, 28] (DLT Points and DLT Lines).

Considering the aspects of finding 2D skeleton points, detecting people, and 3D skeleton pose regression from image, we follow in essence the methodologies developed by Deva Ramanan [31, 6].

2.1 Camera Projection Model and Back-projection

The projection model is a mathematical relation that maps 3D world points onto the camera's image plane. There are several camera models. Below, it is presented the one used in this work. The back-projection is a technique that inverts the projection model by obtaining 3D projection rays, from image pixel 2D.

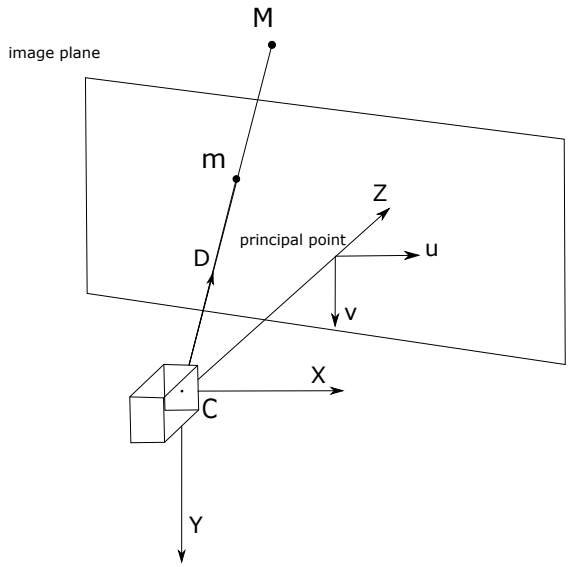


Figure 2.1: Representation of the pin-hole camera model with respective camera frame (u, v), world frame (X, Y, Z) and projection of 2D point in image plane.

2.1.1 Projection Model

The used projection model is the pin-hole model. According to this model, Fig. 2.1, a camera is represented by a single point, pin-hole, and all light beams pass through that point, being then projected onto the image plane. Camera image point is formed by the intersection of the optical ray r, with the image plane, describing a transformation between a 3D Euclidean space and a 2D Euclidean space. A scene point $M = [X\ Y\ Z]^T$ can be mapped to an image point $m = [u\ v]^T$, applying the following equation:

$$u = f\frac{X}{Z}, v = f\frac{Y}{Z} \tag{2.1}$$

where f represents the camera focal length. Using homogeneous coordinates, (2.1) can be rewritten as:

$$m \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} M \tag{2.2}$$

where $\sim$ denotes equal up to a scale factor.

Considering now camera lens' physical aspects, these cause variations in image points coordinates, so they must be considered in the model. Therefore the offset from point C,(c_u, c_v), Fig. 2.1, may cause some translations in image pixels. In addition, since it is desired to obtain

distance in meters instead of pixels, the focal length should also be taken into account,(f_u, f_v). The distortion caused by the non-orthogonality of the horizontal and vertical image axis will be represented by the skew factor, s_k. Considering all these parameters a linear relation between camera coordinates and pixel coordinates in image plane can be obtained through the following matrix:

$$K = \begin{bmatrix} f_u & s_k & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$

Now, considering (2.3) into (2.2) this can be written as:

$$m \sim [K|\, 0]\, M = P\, M \tag{2.4}$$

Since points will be expressed in a different Euclidean coordinate frame, other than the camera's, this can be parametrized by a rigid body transformation (rotation plus translation). Transformation, mapping 3D points from the world to the camera frames, can be defined as:

$${}^{c}M = {}^{c}R_{w}\, {}^{w}M + {}^{c}t_{w} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} {}^{w}M, \tag{2.5}$$

where c stands for camera and w for world coordinate systems. Defining this as the camera's extrinsic parameters, it can be included in (2.4), allowing us to write the camera projection model as:

$$m \sim PM = K\,[R\, t]\, M. \tag{2.6}$$

Knowing projection matrix P, it is possible to characterize camera by decomposing it into extrinsic and intrinsic parameters, see Appendix B.

2.1.2 Back Projection

Having the projection model, it is possible to define a projection ray that, from a given image point, allows us obtain the line that represents all 3D points that are images of such image point. First, we compute the camera's projection center, $C = [X\ Y\ Z]^T$ (Fig. 2.1), starting from the constraint $[0\ 0\ 0]^T = P[C\ 1]^T = [P_{1:3}\ P_4] = [C\ 1] = P_{1:3}C + P_4$. Solving the constraint as a function of C allows us to write the camera projection center as:

$$C = -P_{1:3}^{-1} P_4\ . \tag{2.7}$$

As shown in Fig. 2.1, an image point projection ray will have direction $D = [X\ Y\ Z]$ such that $m \sim P[X\ Y\ Z\ 0]^T$, allowing us to write the previous equation as $m \sim [P_{1:3}\ P_4][D\ 0]^T$ where $D = [X\ Y\ Z]$. Solving as a function of the direction D, we get:

$$D = P_{1:3}^{-1} m\ . \tag{2.8}$$

Finally, the image point back projection ray can be written as:

$$M = C + \alpha D, \alpha \in \mathbb{R}. \tag{2.9}$$

Replacing (2.7) and (2.8) in (2.9), one obtains $M = -P_{1:3}^{-1}P_4 + \alpha P_{1:3}^{-1} m$, which can be rewritten as $R_{proj} = -P_{1:3}^{-1}(-P_4 + \alpha m), \alpha \in \mathbb{R}$, defining a 3D straight line that corresponds to all 3D points that correspond to the same point m on image.

2.2 DLT Based Camera Calibration

The Direct Linear Transformation (DLT) can be described as an algorithm that finds a set of variables, in the form $x_k \sim Ay_k$ for $k = 1, ..., N$. The camera model (2.6), in its essence based on the projection matrix P, can be estimated using correspondences between world and image points in the formalism of a DLT. More precisely, the DLT (developed by Aziz in [2]) obtains the projection matrix P, by solving a linear system on the matrix entries, based on a set of known 3D points in the world frame $\{M_i : M_i = [X_i\ Y_i\ Z_i\ 1]^T\}$ and their images (2D image points) $\{m_i : m_i = [u_i\ v_i\ 1]^T\}$.

In this section we present calibration procedures, as proposed by Manuel Silva et al. in [27]. To make a distinction between the use of points and lines, for calibration, we use the terminology DLT Points and DLT Lines. Below, we detail these methodologies.

2.2.1 DLT Points

Applying the cross product of m_i in both sides of (2.6), $m_i \times m_i = m_i \times (PM_i)$, results in zero in the left hand side of the equation and, thus, $[m_i]_\times PM_i = 0$ where $[m_i]_\times$ denotes the linear cross product operation[1]. Now, considering the properties of the Kronecker product (here

[1]Using this operation, one can write $a \times b = [a]_\times b$ where $[a]_\times$ is a 3×3 skew-symmetric matrix, containing the entrances of the vector a.

denoted as $\otimes$), [25], one can obtain an equation factorizing the data and variables as:

$$(M_i^T \otimes [m_i]_\times) vec(P) = 0, \tag{2.10}$$

where $vec(P)$ represents the column vectorization of the matrix P, i.e. formed by stacking all the columns into a single vector. Then, for a set of pairs $\{M_i, m_i\}$, for $i = 1, \ldots N$, using (2.10) we have:

$$\begin{bmatrix} M_1 \otimes [m_1]_\times \\ \vdots \\ M_N \otimes [m_N]_\times \end{bmatrix} \begin{bmatrix} P_{11} \\ P_{12} \\ \vdots \\ P_{34} \end{bmatrix} = 0, \tag{2.11}$$

which can be rewritten as:

$$\underbrace{\begin{bmatrix} M_1 & 0 & -u_1 M_1 \\ 0 & M_1 & -v_1 M_1 \\ M_2 & 0 & -u_2 M_2 \\ 0 & M_2 & -v_2 M_2 \\ \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots \\ M_N & 0 & -u_N M_N \\ 0 & M_N & -v_i M_N \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} P_{11} \\ P_{12} \\ \vdots \\ P_{34} \end{bmatrix}}_{\times} = 0. \tag{2.12}$$

Each of the N entries provides a set of three equations in the entries of $vec(P)$. However, only two of them linearly independent.

The system presented in (2.12) can be solved using a single value decomposition of A, allowing to obtain the projection matrix P, for a minimum of 6 pairs of points matches (i.e. $N = 6$). However, a pre-normalization of the input data is crucial on the implementation of this algorithm. For that purpose, we use the method proposed by Hartley, in [15]. Hartley suggested an appropriate transformation to translate all data points (3D and 2D points) so that: 1) their centroids are at the origin; and 2) the average distance of data points to the origin is equal to$\sqrt{2}$ for image points and$\sqrt{3}$ for 3D points.

DLT-Points with Radial Distortion

As noted by Fitzgibbon [10], true lens distortion curves are typically very complex to represent, requiring for that reason the use of high-order models (or lookup tables) to model camera radial distortion effect with high precision. On the other hand, considering typical computer vision applications, accuracies of the order of a pixel are all that are required. Therefore, an approximation to the cameras' true distortion functions should perform as well as the preciser ones. The division model for radial distortion can be defined as, $m_u = m_d/(1 + \lambda||m_d||^2)$, where λ represents the distortion parameter. It can be rewritten in homogeneous coordinates as:

$$\begin{bmatrix} u_u \\ v_u \\ 1 \end{bmatrix} \sim \begin{bmatrix} u_d \\ v_d \\ 1 + \lambda||m_d||^2 \end{bmatrix}, \tag{2.13}$$

which implies that an undistorted point is a simple function of a distorted point $m_u = m_d + \lambda||e_d||$ where $e_d = [0\ 0\ ||m_d||]^T$. Now, adding radial distortion to (2.11) allows us to rewrite the system of equations as:

$$\begin{bmatrix} M_1 \otimes [m_{1d} + \lambda e_{1d}]_\times \\ \vdots \\ M_i \otimes [m_{id} + \lambda e_{id}]_\times \end{bmatrix} \begin{bmatrix} P_{11} \\ P_{12} \\ \vdots \\ P_{34} \end{bmatrix} = 0, \tag{2.14}$$

or $(A_{i1} + \lambda A_{i2})vec(P) = 0$ where $A_{i1} = M_i^T \otimes [m_{id}]_\times$ and $A_{i2} = M_i^T \otimes [e_{id}]_\times$ which can be solved as polynomial eigenvalue problem, [3], i.e.:

$$(A_1^T A_1 + \lambda A_1^T A_2)vec(P) = 0 \tag{2.15}$$

where $A_1 = M_i^T \otimes [m_{id}]_\times$ and $A_2 = M_i^T \otimes [e_{id}]_\times$ for the i pairs (M_i, m_i), obtaining projection matrix P and distortion parameter λ.

2.2.2 DLT Lines

In this section, it is presented the calibration methodology for DLT-Lines, including the estimation of radial distortion, from [28]. Given a 3D line L_i, its projection on the camera image plane l_i can be represented by the cross product of two image points in projective coordinates, $l_i = m_{1i} \otimes m_{2i}$. Any point m_{ki}, lying in the line l_i, implies that $l^T\ m_{ki} = 0$. Applying the

multiplication by l^T on both sides of (2.6), i.e., $l^T\ m_{ki} = l^T\ PM_{ki}$, leads to:

$$l_i^T P M_{ki} = 0, \tag{2.16}$$

where M_{ki} is a 3D point in projective coordinates, lying in L_i. As in the case of DLT-Points, using the Kronecker product, one can obtains a factorizing form (vectorized projection matrix) as:

$$(M_{ki} \otimes l_i^T) vec(P) = 0. \tag{2.17}$$

Each pair (M_i, L_i) allows us to write a constraint on the form of (2.17). So, in order to determine the twelve entries of matrix P, 12 pairs of matches (M_i, L_i) will be needed.

DLT Lines with Radial distortion

Also, applying radial distortion model (as in 2.2.1) to a line results in:

$$l_{12} = \begin{bmatrix} u_{1d} \\ v_{1d} \\ 1 + \lambda s_1^2 \end{bmatrix} \times \begin{bmatrix} u_{2d} \\ v_{2d} \\ 1 + \lambda s_2^2 \end{bmatrix} = l_{12} + \lambda e_{12}, \tag{2.18}$$

where s_i is the norm of distorted point $s_i^2 = u_i^2 + v_i^2$. $l_{12} = \begin{bmatrix} u_{1d}\ v_{1d}\ 1 \end{bmatrix}^T \times \begin{bmatrix} u_{2d}\ v_{2d}\ 1 \end{bmatrix}^T$ and there is a distortion correction term $e_{12} = \begin{bmatrix} v_{1d}s_{22} - v_{2d}s_{21}\ u_{2d}s_{21} - u_{1d}s_{22}\ 0 \end{bmatrix}^T$.

Now, applying the point-to-line constraint allows us to write:

$$\begin{bmatrix} M_1 \otimes [l_{1d} + \lambda e_{1d}]_\times \\ \vdots \\ M_N \otimes [l_{Nd} + \lambda e_{Nd}]_\times \end{bmatrix} \begin{bmatrix} P_{11} \\ P_{12} \\ \vdots \\ P_{34} \end{bmatrix} = 0, \tag{2.19}$$

which can also be solved using polynomial eigenvalue solver, as in 2.2.1:

$$(B_1^T B_1 + \lambda B_1^T B_2) vec(P) = 0, \tag{2.20}$$

where $B_{i1} = M_i^T \otimes [l_{id}]_x$ and $B_{i2} = M_i^T \otimes [e_{id}]_\times$, obtaining projection matrix P and a distortion parameter λ, from $(B_1^T B_1 + \lambda B_1^T B_2) vec(P) = 0$.

The main advantage of DLT lines over DLT points is the possibility of applying line fitting and finding techniques, that will add more robustness to user error inputs.

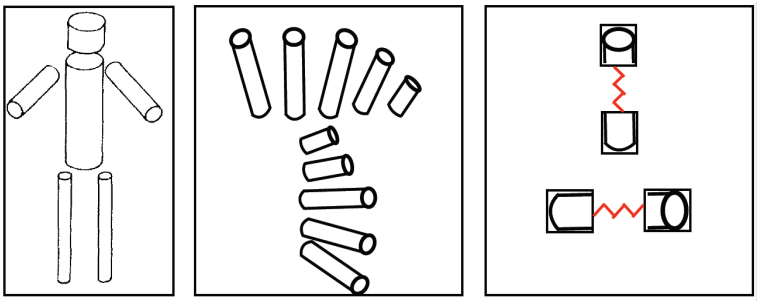


Figure 2.2: The classic articulated limb model of Marr and Nishihara, [26] . In the middle, the different orientation and foreshortening states of a limb, each of which is evaluated separately in classic articulated body models. On the right, these transformations with a mixture of non-oriented pictorial structures, in this case tuned to represent near-vertical and near-horizontal limbs, from [31].

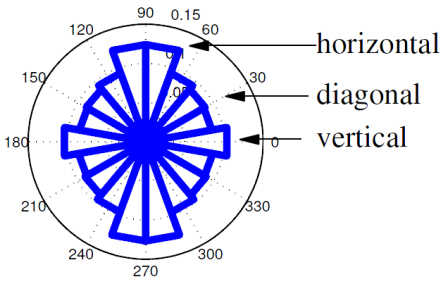


Figure 2.3: Average HOG feature, from [31], as a polar histogram over 18 gradient orientation channels as computed from the PASCAL 2010 dataset. On average, images contain more horizontal gradients than vertical gradients, and much stronger horizontal gradients as compared to diagonal gradients.

2.3 Human Pose Estimation

For 3D Human pose estimation, it is first performed a 2D pose detection estimate (based on Histogram of Gradients [31]) and, then, the 2D model parts from image, followed by a matching with a library of 3D model poses [6]. Below, these approaches are detailed.

2.3.1 Image based 2D pose estimate

2D pose estimation, presented in [31], uses a novel representation of model parts where, instead of using articulated oriented limb parts, it approximate the model to non-oriented parts. The goal is to represent near-vertical and near-horizontal limbs, see Fig. 2.2. On the other hand, the orientation can be captured on statistics of background features, [8], see Fig. 2.3.

So, for an image, the pixel location of part i, $p_i = (x, y)$, the mixture component of part i

can be written $i \in \{1, ..., K\}, p_i \in \{1, ..., L\}, t_i \in \{1, ..., T\}$, being t_i the type of part i (e.g. a human hand). First, to score a configuration of parts, it is defined a compatibility for type of parts that factors into a sum of local and pairwise scores:

$$S(t) = \sum_{i \in V} b_i^{t_i} + \sum_{ij \in E} b_{ij}^{t_i, t_j}. \tag{2.21}$$

The parameter $b_i^{t_i}$ favors particular type assignments for part i, while the term $b_{ij}^{t_i,t_j}$ favors particular co-occurrences of part types. For example, if part types correspond to orientations and part i and j are on the same rigid limb, then $b_{ij}^{t_i,t_j}$ would favor consistent orientation assignments. A K-node relational graph can be written, $G = (V, E)$, where edges specify which pair of parts are constrained.

A full score associated with a configuration of part types and position can be written as:

$$S(I, p, t) = S(t) + \sum_{i \in V} w_i^{t_i} \cdot \phi(I, p_i) + \sum_{ij \in E} w_{ij}^{t_i, t_j} \cdot \psi(p_i - p_j), \tag{2.22}$$

where $\phi(I, p_i)$ is a feature vector extracted from pixel location, such as the one in Fig. 2.3, p_i in image I, and $\psi(p_i - p_j) = [dx\ dx^2\ dy\ dy^2]^T$, where $dx = x_i - x_j$ and $dy = y_i - y_j$, the relative location of part i with respect to j. This relative location is defined with respect to the pixel grid (and not the orientation part).

The first sum in (2.22) is an appearance model, that computes the local score of placing a template. The second term can be interpreted as a "switching" spring model, that controls the relative placement of part i and j by switching between a collection of springs. Each spring is tailored for a particular pair of types $(t_i; t_j)$, and it is parametrized by its rest location and rigidity, which are encoded by $w_{ij}^{t_i,t_j}$. Maximizing S from (2.22) over p and t, on a given graph G, allows us to compute the message part i by the following:

$$score(t_i, p_i) = b_i^{t_i} + w_{t_i}^i \cdot \phi(I, p_i) + \sum_{k \in kids(i)} m_k(t_i, p_i) \tag{2.23}$$

$$m_i(t_i, p_i) = \max_{t_i}\ b_{ij}^{t_i,t_j} + \max_{p_i}\ score(t_i, p_i) + w_{ij}^{t_i,t_j} \cdot \phi(p_i - p_j). \tag{2.24}$$

While: 1) (2.23) computes the local score of part i, at all pixel locations p_i and for all possible types t_i, by collecting messages from the children of i; 2) (2.24) computes, for every location and possible type of part j, the best scoring location and type of its child part i. Once the messages are passed to the root part, $(i = 1), score1(c1; p1)$ represents the best scoring configuration for each root position and type. One can use these root scores to generate multiple

detections in image I by thresholding them and applying a non-maximum suppression (NMS). By keeping track of the argmax indices, one can backtrack to find the location and type of each part, in each maximal configuration.

2.3.2 3D model matching

For 3D pose estimation, since big sets of 3D poses are available (that makes it possible to predict 3D poses from 2D), using 2D members and joints positions, 3D pose can be estimated based on matching from [6].

A probability $p(M|m)$ is modeled with a non-parametric nearest neighbor model. Assuming that we have a library of 3D poses $[M_i]$, paired with a particular camera projection $[P_i]$ such that the associated 2D poses are given by $P_i(M_i)$, the probability distribution over 3D poses based on re-projection error becomes

$$p(M = M_i|m) \propto e^{-\frac{1}{\sigma^2}\|P_i(M_i)-m\|^2}, \tag{2.25}$$

where the MAP estimate is given by the 1-nearest neighbor method. Then squared re-projection error can, then, be reduced to:

$$P_i^* = argmin \|P(M_i) - m\|^2 . \tag{2.26}$$

A short list of k candidates is built according to (2.25). These k candidates can be re-sorted, according to the camera matrix. Since we know its intrinsic parameters, and the corresponding 2D and 3D pose for best score candidates, it is possible to estimate camera rotation that will align 3D points with their 2D respective projections Having the best candidate, simply replace the 3D coordinates (X_i, Y_i) by their scaled 2D counterparts (u, v) under a weak perspective model:

$$M^* = [su\ sv\ Z_i], where\ s = \frac{average(Z_i)}{f}, \tag{2.27}$$

where f is focal length, given by the camera's intrinsic in P_i, and (Z_i) is the average depth of the 3D joints.

Chapter 3

Camera Calibration

In order to extract measurements from image, we first need to calibrate camera to obtain a linear relation between world points and their projection onto the image, so that later one can obtain a metric information of the world through the image. Given a camera projection model and the methodologies to compute its parameters, in this section it is firstly explained how the environment is set up, in order to perform a camera calibration using previously described algorithms, aided by depth color camera, RGBD. It will also be presented the developed methods that use Direct Linear Transformation (DLT) techniques, with SIFT algorithm [24], in order to automate the calibration process (making it easier for the user).

3.1 Color-Depth Auxiliary Camera

The most common camera calibration methods are based in imaging a checkerboard pattern, in various (distinct) 3D poses. In these methods, camera parameters are estimated by solving a homogeneous linear system, that captures the homography relation between multiple perspective images of the same plane. A RGBD camera can easily provide, with some precision, the coordinates of a set of 3D points required to solve a single linear system (DLT calibration).

For the different calibrations methodologies of an RGB camera, the following setups are used. In the Bouguet calibration technique, Fig. 3.1, a chess board in multiple poses is used, while in our setup a color depth camera, Figure 3.2, provides the required 3D information to solve camera calibration problem.

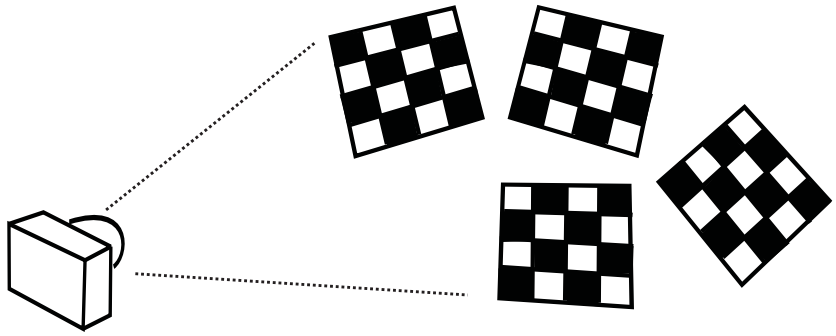


Figure 3.1: Bouguet calibration setup. A checkerboard is imaged at various poses.

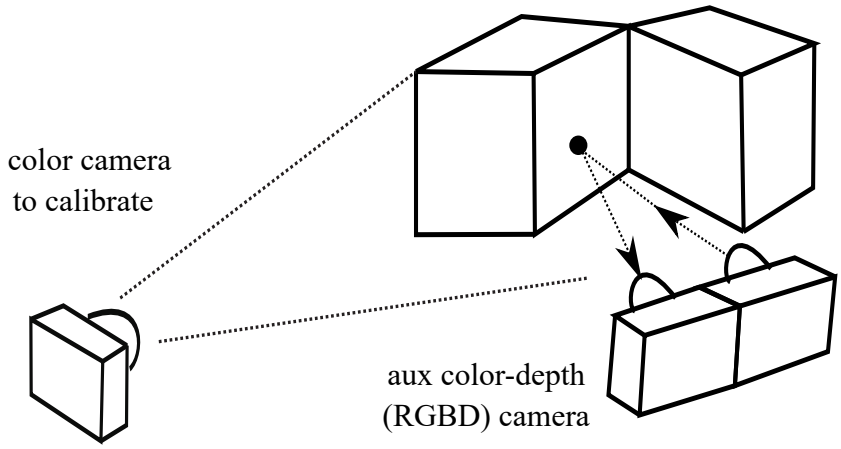


Figure 3.2: Calibration helped by a color-depth (RGBD) camera.

3.2 Calibration Methods using Color and Depth Data

In the following sections, we detail camera calibration methods based on image and 3D data. In our work, camera calibration is used to take metric measurements. Given that the image data lacks depth information, we combine back-projection with a ground plane constraint, hence obtaining 3D data for ground points. In order to simplify using the ground plane constraint, our calibration methods rely on setting the camera extrinsic parameters with respect to a ground plane reference frame (details in Sec. 3.2.1).

Calibration methods proposed in this section are make a direct to the use of data provided by the user, points or lines, in the both the image and the 3D spaces. Later, we propose procedures for automating the calibration methods. Automation is mostly designed to reduce the amount of user input, making the calibration methods easier and faster.

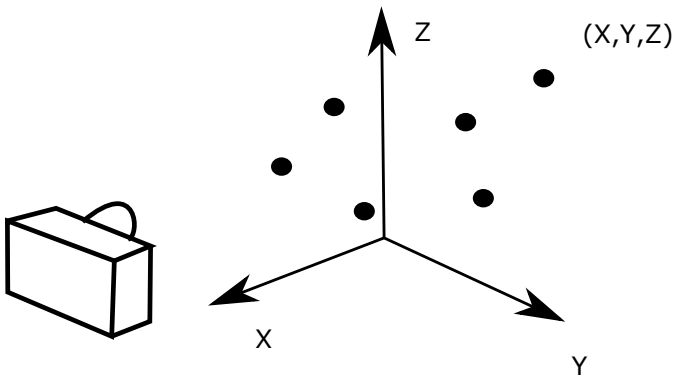


Figure 3.3: DLT-Points setup. Points in 3D are matched with image points in 2D to obtain camera calibration.

3.2.1 Set 3D Data to a Ground Plane Reference Frame

The ground plane reference frame ($z = 0$) is used as coordinate system for the 3D data (Kinect / RGBD). For that purpose, after having a 3D pointCloud, a minimum of 3 points is picked on the floor and a least squares method is performed to find the best fitting plane. With the plane defined, two points of it can be picked to obtain the new desired frame to define 3D data: 1) one point for origin of frame; and 2) one point for X or Y axis, being Z axis the normal vector to the plane, applying then (2.5) to transform frame. Consequently the camera calibration results will have extrinsic parameters w.r.t. the ground plane.

3.2.2 DLT-points based Calibration

In this method, in order to estimate the projection matrix P from (2.6), a set of points chosen by the user in the 2D projective plane (RGB image) and the corresponding points in 3D projective space (in depth-color image) provide the required data to solve (2.14) and the respective polynomial eigenvalue problem (2.15). Allowing us to obtain one estimate for the camera's calibration parameters (i.e. its projection matrix P that makes the matching between the points in 3D with the ones in image), Fig. 3.3.

3.2.3 DLT-lines based Calibration

In this approach, lines provided by the user in 2D (RGB image) and in 3D (RGBD data) allow to solve (2.20). Like it was said before, the main advantage of using lines over points is the possibility of applying techniques for line fitting, in order to improve the results. So, when points defining line are picked by the user, line fitting techniques are applied. RANSAC technique

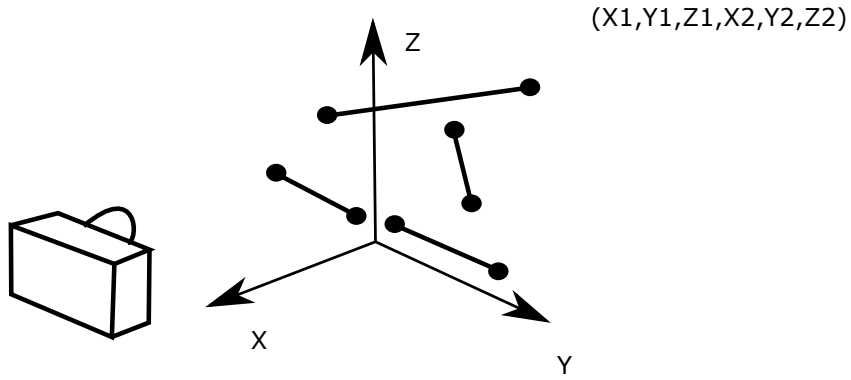
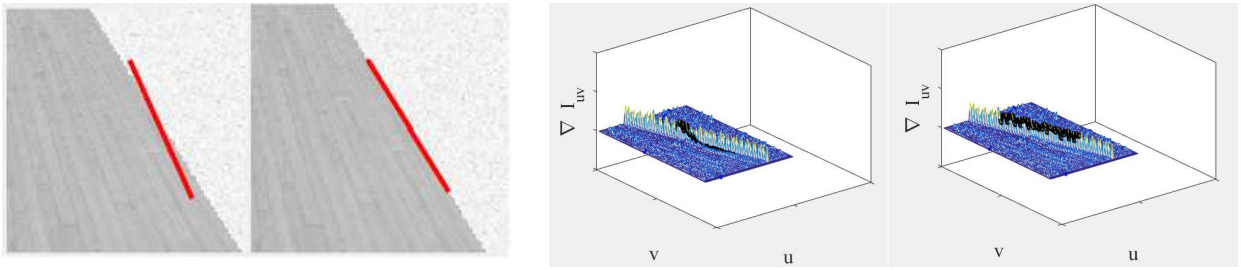


Figure 3.4: DLT-Lines setup. Fitted Lines in 3D are matched with tuned image lines in 2D to obtain camera calibration.



(a) 2D line fitting using as input 2D user input line.

(b) 2D line in black fitting to image edge.

Figure 3.5: Methodology for picking Lines helping in user input. Best line configuration is found by minimizing the cost function $C = -\sum_k \|\nabla I(m_k)\|$.

for 3D data (from depth image) and for 2D data are picked by the user, points defining a line are adjusted to the nearest line in image (as shown in Fig. 3.5), reducing the user input error. Fig. 3.4 shows the DLT-Line based Calibration setup.

3.3 Assisted Matching of Lines

Since the clicked points will have about 1 pixel uncertainty, this method tries to avoids that, by allowing to pick matching lines found by local gradients maximums. It will remove the click error, while making easier and faster for the user.

This method can be described as follows. First run Canny Edge detector and Hough Transform to detect lines in RGB; and, then, in depth-color image, run line fitting techniques (see Fig. 3.5). After that, it is requested that the user perform the match between the fitted lines in RGB and depth-color image. In the 3D points obtained from depth map, a RANSAC technique

is used (see Fig. 3.7). Solving (2.17) allows to estimate projection matrix P (see Fig. 3.6).

3.4 Automated Calibration

The number of lines provided for the user to click on might not be enough to estimate camera projection matrix. In order to solve this, the method concatenates some equations from SIFT matching points between images (2.11), with (2.17). SIFT results will provide (M_i, m_i) one projection matrix estimate. This estimate will try to automatically make line correspondence (M_i, l_i), making it easier for the user (since it won't require any user input).

This method uses the following steps. It starts by using SIFT algorithm, in order to detect matching points between RGB (m_i) and depth-color (M_i) images. Using (2.11) from DLT-Points calibration, and matching of points from previous step is computed, in order to perform a first estimate of projection matrix P. In depth-color image, through Canny Edge detector and Hough Transform methods, lines are detected. Fitting and filtering processes are applied to those lines, as well as a RANSAC algorithm to the 3D points. Using the first estimate of projection matrix P, it is then possible to obtain the matching between the lines found in depth-color image that lie in RGB image. Solving the system (2.17) concatenated with 2.11 allows us to perform a new estimate of projection matrix P, Fig. 3.8

Algorithm 1 Automated Calibration

Detect and match SIFT features in RGB and depth-color images
Estimate projection matrix P of the RGB camera
Detect lines in depth-color image using Canny Edge detector and Hough Transform
Fitting and filtering is applied to found lines
P estimate is used to match lines between RGB and RGBD
Reestimate projection matrix P using matching lines and SIFT results

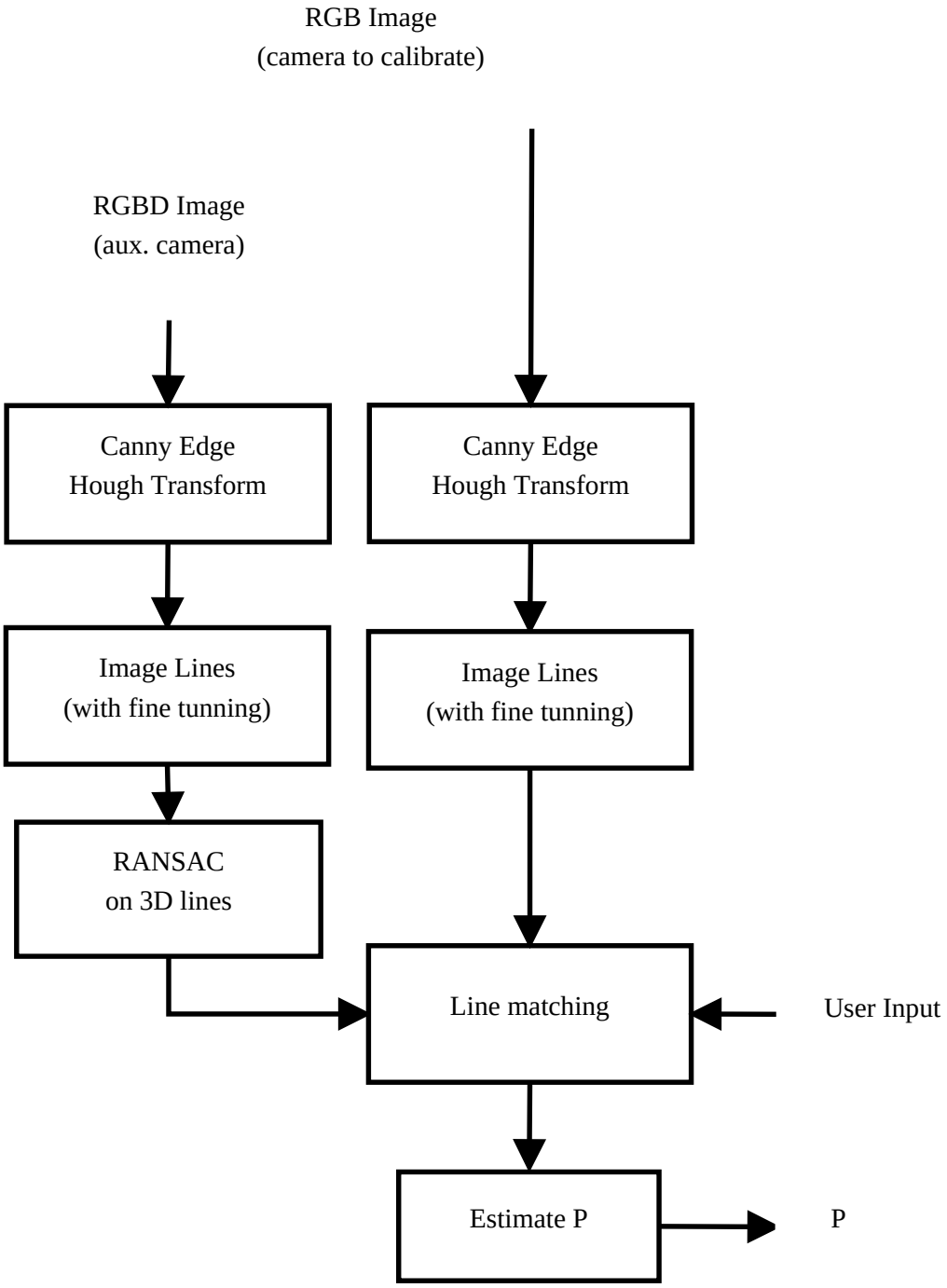


Figure 3.6: Assisted Matching of Lines. Methodology where user input is required to perform matching between 3D and 2D lines detected. DLT-Lines is then used to perform camera calibration estimate.

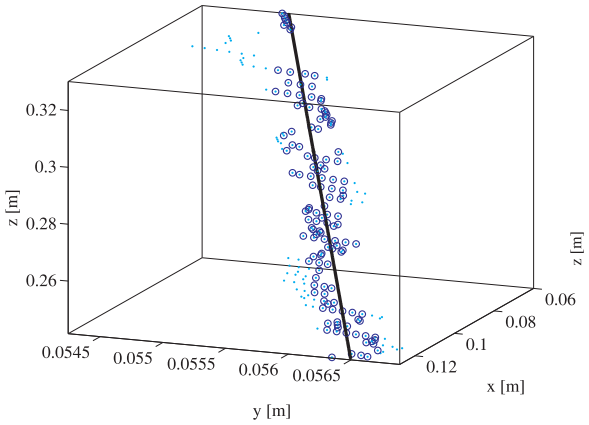


Figure 3.7: 3D line fitting using RANSAC.

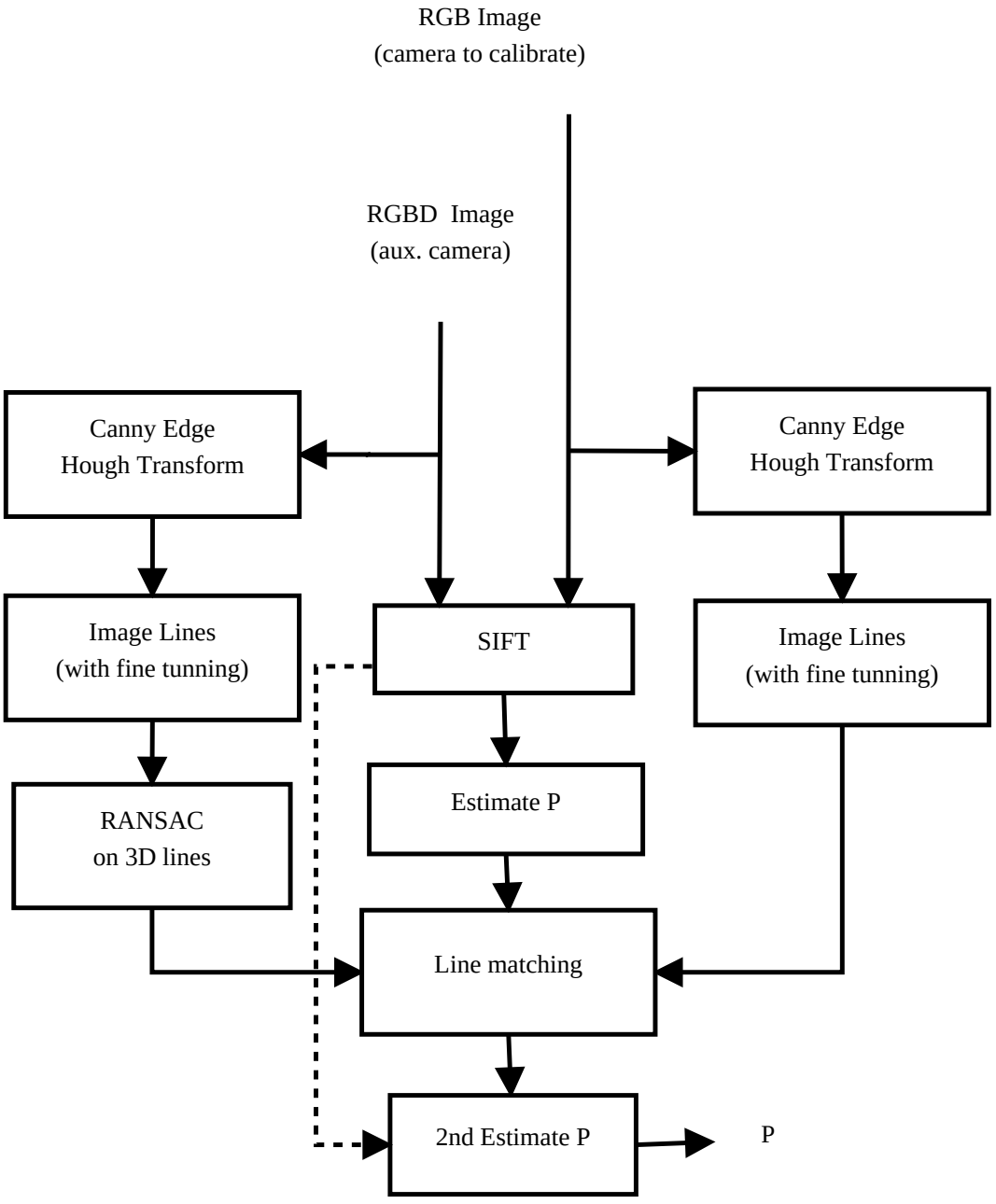


Figure 3.8: Automated Calibration. Camera Calibration methodology using SIFT to obtain a 1st estimate which will help in line matching between images in order to perform calibration.

Chapter 4

Height Measurement Methodologies

Having a calibrated camera, it is now possible obtain measurements from its images. In this section, those measures techniques are described. Techniques for obtaining vertical measures of, for example, the height of a person. It is also presented a technique to estimate the height of a person not standing in vertical. Such methodologies make use of clicked points in image, as sources for back-projections rays that will help in performing such calculations.

4.1 Measuring the Height of a Person

The technique presented in this section relies on the fact that the person is standing, and one knows its position on the floor. It is, then, possible to estimate its height. Since in a 2D image (from a RGB camera) the depth information is lost, we need to back-project image points and intersect them with any known 3D information (in this case the person's feet are on the floor).

In order to estimate a person height, it is first performed a back-projection of the first user clicked point into the floor, where person's feet are (see Fig. 4.1).

4.1.1 Ground point between feet and optic ray tangent to the head

For a calibrated camera, an image point can be mapped into a back-projection ray in the world. So, knowing the floor point back projection ray, and since it is known that point lies on the floor (this is $Z = 0$ in the world frame), it is possible to find the $(X,\ Y)$ coordinates of image clicked point on the world. More precisely:

$$M_{floor} = [X\ Y\ Z = 0]^T = f(m, \alpha), \tag{4.1}$$

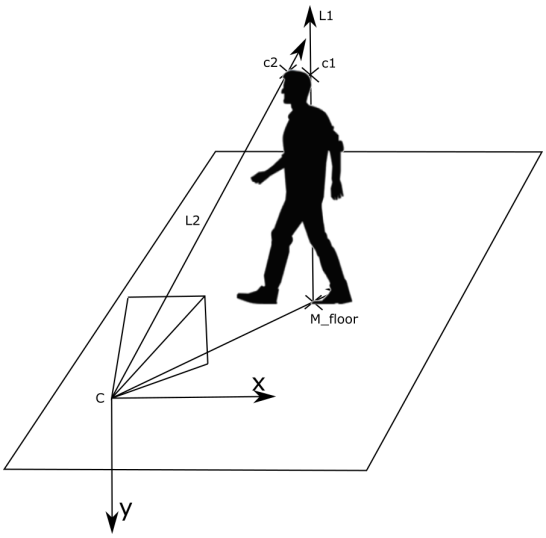


Figure 4.1: Back-projection line of user clicked on point on floor with respective projection on vertical, L1. While L2 is the back-projection line of user clicked point in head. Closest point in each one of them is respectively c1 an c2.

where f is the linear function (back-projection ray) defined in sec.2.1.2, m is the user clicked point and α is the chosen scalar that will make M having $Z = 0$. Using another user clicked point, on the top of head, it is possible to obtain optic ray tangent to the head, L_2, as shown in Fig. 4.1.

4.1.2 Height estimation

Since it is assumed that the person is standing up, it is possible to define a vertical line from M_{floor} using vector normal to the floor, in this case $[0\ 0\ 1]^T$:

$$L_1 = M_{floor} + t \cdot [0\ 0\ 1]^T, \quad t \in \mathbb{R}, \tag{4.2}$$

and intersect it with the projection ray from user clicked point on head, i.e.:

$$L_2 = C + s \cdot B_{ray}, \quad s \in \mathbb{R} \tag{4.3}$$

where $B_{ray} = P_{1:3}^{-1} m_{head}$ defines the back-projection ray, being m_{head} the clicked point on head. Since these two lines do not intersect, we get the points where their distance is minimum, c_1 and c_2 as shown in Fig.4.1.

In order to find the point where L_1 is closest to L_2 it is first performed the cross product between the two lines direction vectors: $N = [0\ 0\ 1]^T \times B_{ray}$, obtaining a vector N orthogonal

to both lines. The plane formed by the translations of L_2 along N contains the point C and it is perpendicular to $n_2 = d_2 \times N$, where d_2 is the directional vector of L_2.Therefore, the intersecting point of L_1 with the above-mentioned plane, which is also the point on Line 1 that is nearest to Line 2, is given by:

$$c_1 = M_{floor} + \frac{(M_{floor} - C)^T \cdot n_2}{d_1^T \cdot n_2} d_1. \tag{4.4}$$

Similarly, the point on Line 2 that is nearest to Line 1 is given by:

$$c_2 = C + \frac{(C - M_{floor})^T \cdot n_1}{d_2^T \cdot n_1} d_2, \tag{4.5}$$

where $n_1 = d_1 \times N$, being d_1 directional vector of L_1. Knowing where the point lies on the L_2, it is possible to estimate the person height by that third coordinate, "Height" of the mid point between c_1 and c_2:

$$c_{midpoint} = \frac{c_1 + c_2}{2}. \tag{4.6}$$

4.1.3 Height Estimation Method Summary

In short, assuming: (i) Person *standing up* (vertical pose) on the ground; (ii) The *ground is planar*; (iii) The camera is *calibrated w.r.t. the ground plane*[1], the height estimation method can be resumed in the following steps:

1. Image point between the *person's feet* is used to define a *3D ground point* (4.1);
2. Image point tangent to the *top of the head* is used to define a backprojection optical ray (4.3);
3. The intersection of the head tangent ray with a vertical line from the middle point of the feet defines the 3D point indicating of top of the head (4.6).

4.2 Non-Vertical Body Pose

This section considers the case where a person is in a non-vertical pose, e.g. in a sitting position. As in the previous case we want to perform an estimate of his height, given an RGB

[1] In particular the extrinsic parameters of the camera are defined such that the ground plane is characterized by $z = 0$. C.f. section 3.2.1.

image. Despite the fact that the developed methodologies for finding a person in a image[2] and estimating its skeleton pose[3], [31, 6], Sec. 2.3.1,2.3.2, are already very effective, it will only provide results in non-metric units, Fig.4.4(a). In 3D pose estimation it is used a database of normalized skeleton poses to find best matching one, with the points detected in image.

Since we want to estimate a person height we need valid metric units in the 3D skeleton pose. For that in our work we will scale the skeleton according to the depth of person in picture. In this section it will be described how this 3D pose estimate solution was developed in order to provide the needed results in metric units, to perform an person height estimate in non-vertical pose.

To explain the complete method, first let us consider three cameras:

1. Toolbox First Camera, $P = K[R\,t]$, relates 2D pose image points with the ones in the 3D skeleton database;

2. Toolbox Aligned Camera, relates 3D skeleton points with their *aligned* image points projections, $P_c = K[I\ 0]$;

3. Real camera who took the photo, $P_w = K[R_w\ t_w]$;

All three cameras have the same intrinsic parameters, namely the intrinsic parameters estimated for the real camera, K. The first two cameras consider normalized data, $\widetilde{M}$, while the third one considers real 3D metric data, $M = \alpha\widetilde{M}$, see Fig. 4.2.

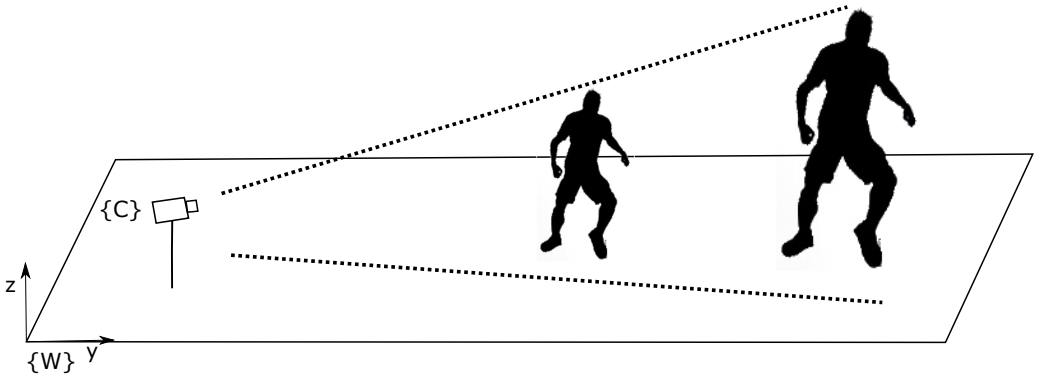


Figure 4.2: 3D pose model estimation from scaled normalized body $\widetilde{M}$. C stands for aligned camera frame while W stands for world frame where real camera projection, P_w is valid.

[2]2D person detection package in `https://github.com/dykestra/Yang_Ramanan`
[3]3D pose matching in `https://github.com/flyawaychase/3DHumanPose`

4.2.1 Up-to Scale Pose

The used skeleton find toolbox provides an up-to scale 3D pose. It starts by determining 2D person position in picture, using the techniques described in 2.3.1. This method uses a model with flexible mixtures of parts and an histogram of gradients, that will allow us to obtain possible person pose detection, with respective scores 2.22, giving the best 2D members pose detection in an image and, as a consequence, the 2D person joints locations.

Given the 2D joint locations, it is then estimated a list of k best possible poses candidates, from the 3D skeleton database, using nearest-neighbors method, Sec. 2.3.2. At the same time is estimated the pose of the first camera, P, for each of the candidates. The best 3D pose is estimated by minimizing (2.26), in (2.25).

The best skeleton pose candidate obtained can then be aligned with their image projections points. In other words, the skeleton pose candidate is rigidly transformed to the camera coordinate system, which allows obtaining a simplified camera P_c independent of the coordinate system where the skeleton was defined:

$$m \sim K[R\,t][\widetilde{M}\ 1]^T \rightarrow m \sim K[I\ 0][^c\widetilde{M}\ 1]^T \quad . \tag{4.7}$$

The toolbox provides as a final result an up-to scale model $\{(m, \widetilde{M})\}$. The resulting pose is *"visually correct"* but still needs a correcting scale factor. Figure 4.4(a) illustrates a normalized body size being scaled to the right size. In the next subsection it will be proved the validity of applying such scale factor, and how to obtain it in order to estimate metric 3D pose.

4.2.2 Metric 3D Pose

To scale the 3D skeleton structure to valid metric values, we get the distance to one skeleton point, and we scale it according to such distance.

Firstly, let us say we have a calibrated camera P_w and know the coordinates of a 3D human-body point in the world, using techniques previous presented based on one user clicked point on the floor, to get to ${}^wM$ (with (4.1)). One can then obtain the distance of ${}^wM$ to the camera center,${}^wC$ as $\|{}^wM - {}^w\,C\|$.

Let us say now that we know ${}^c\widetilde{M}$, the 3D coordinates of the same point provided by the toolbox, where such point is represented in metric coordinates as $\alpha{}^c\widetilde{M}$ and α is the scaling factor. Since the toolbox considers the coordinate system coincident with the projection center,

and no rotation exists, i.e. $P = K[I\ 0]$, then we can state

$$\alpha \|{}^c\widetilde{M}\| = \|{}^wM - {}^wC\| \tag{4.8}$$

and so

$$\alpha = \frac{\|{}^wM - {}^wC\|}{\|{}^c\widetilde{M}\|}. \tag{4.9}$$

The toolbox in its most direct usage returns a minimized set of data. More in detail, instead of returning $\{(m_i, {}^c\widetilde{M}_i)\}$, where $m_i \in P^2$ (the projective plane) and ${}^c\widetilde{M}_i \in \mathbb{R}^3$ it returns $\{(H^{-1}(m_i), Z_{DS}(i)\}$, where $H^{-1}(m_i) \in \mathbb{R}^2$, $Z_{DS}(i) \in \mathbb{R}$ and H^{-1} denotes dehomogenization, e.g. $H^{-1}([a\ b\ c]^T) = [a/b\ a/c]$, a function inverting homogenization $H([a\ b]^T = [a\ b\ 1]^T)$. In the following we deduce the expressions based on the output of the toolbox.

First let us define the 3D metric data from the Real Camera as ${}^cM = [X_{cam}\ Y_{cam}\ Z_{cam}]^T$, and the data from the 3D skeleton dataset be ${}^c\widetilde{M} = [X_{DS}\ Y_{DS}\ Z_{DS}]^T$. Then $Z_{cam} = [0\ 0\ 1]{}^cM$ and $Z_{DS} = [0\ 0\ 1]{}^c\widetilde{M}$, therefore (4.9) can be rewritten as

$$\alpha = \frac{\|{}^cR_w^wM + {}^ct_w - {}^cC\|}{{}^c\widetilde{M}} = \frac{\|{}^cM\|}{\|{}^c\widetilde{M}\|}. \tag{4.10}$$

Instead of norms we can use vectors, $\alpha {}^c\widetilde{M} = {}^c M$. We can just consider third coordinate(3rd equation), as $\alpha[0\ 0\ 1]{}^c\widetilde{M} = [0\ 0\ 1]{}^cM$

$$\alpha = \frac{[0\ 0\ 1]\widetilde{M}}{[0\ 0\ 1]M} = \frac{Z_{cam}}{Z_{DS}} \tag{4.11}$$

where Z_{DS} is 3rd coordinate of dataset used, `Human3.6M`, and the distance of our person foot to the camera (with (4.1)), is Z_{cam}, see Fig. 4.3. In the following we denote Z_{DS} as the Z coordinate of a control point of the skeleton, e.g. a point in-between the foot, and we denote $Z_{DS(i)}$ a generic point of the skeleton.

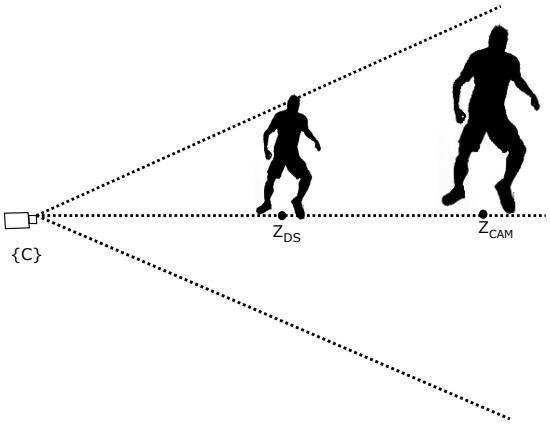


Figure 4.3: 3D pose model estimation using aligned camera frame, C. Z_{DS} stands for 3rd coordinate in camera frame in normalized body, while Z_{cam} stands for 3rd coordinate in real camera, which means distance from camera center to person foot.

All points can now be scaled

$$M_i = \alpha \widetilde{M_i} = \frac{Z_{cam}}{Z_{DS}} \widetilde{M_i} \tag{4.12}$$

where α is a constant scale. The toolbox only provides Z_{DS}, the 3rd coordinate. We still need to convert to $\widetilde{M_i}$ into m_i and Z_{DS}

$$\widetilde{M_i} = H(H^{-1}(K^{-1} \cdot m_i)) \cdot Z_{DS(i)} \quad . \tag{4.13}$$

Applying the scale factor α to previous equation

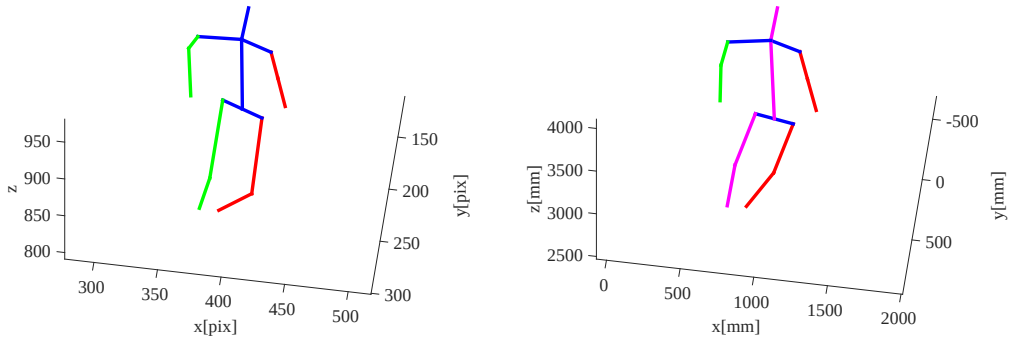
$$M_i = \alpha H(H^{-1}(K^{-1} \cdot m_i)) \cdot Z_{DS(i)} \, . \tag{4.14}$$

Finally, the expression to obtain scaled structure, in metric units, is

$$M_i = \frac{Z_{cam}}{Z_{DS}} H(H^{-1}(K^{-1} \cdot m_i)) \cdot Z_{DS(i)}. \tag{4.15}$$

4.2.3 Summary of Height Estimation for Non-Vertical Poses

In short, assuming: (i) 2D joints locations are correctly recognized in image; (ii) reprojection of pose, given the camera intrinsic parameters, that allows the selection of a non-ambiguously pose; (iii) possible to setmatch an image point to 3D coordinates (in the previous section we considered the middle point of the feet on the ground).



(a) 3D pose model in non-metric unities (b) 3D pose model in metric unities

Figure 4.4: 3D pose model estimated in (a) using (2.27) for a scaled z, while in (b) it is used (4.15) providing metric units, making it possible to estimate height by adding all magenta segments.

The method can be described as:

1. Estimate the 3D location of the middle point between the feet through backprojection;
2. Compute camera projection center
3. Compute distance to camera $L = \|{}^{w}M - {}^{w}C\|$ and then scaling $\alpha = L/\|{}^{w}\widetilde{M}\|$
4. Obtain the skeleton in metric coordinates as $M_i = \alpha^c \widetilde{M}_i, i \in 1, \ldots, N$
5. Estimate the height as

$$h = \sum_{(i,j)\in S} \|M_j - M_i\|$$

where S denotes the set of skeleton links, encompassing leg length, trunk and head length, see Fig. 4.4(b).

Chapter 5

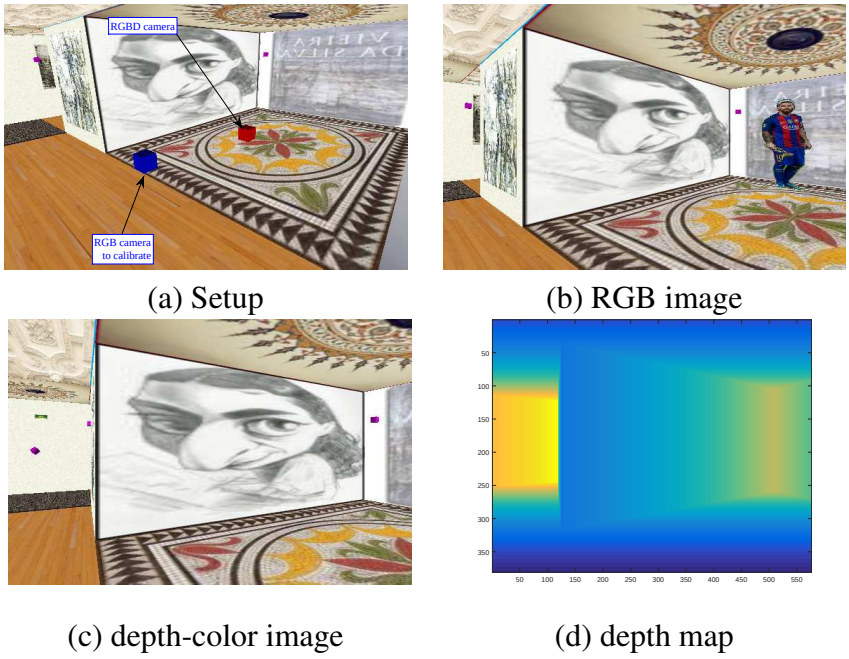
Experiments and Results

Having now a calibrated camera and defined methodologies for obtaining measures from images, it is possible to perform some tests, in order to evaluate the proposed methods. In this section, the calibration and height measurements methodologies are tested using simulated and real data. The simulated setup is built using the Matlab Virtual Reality Modeling Language toolbox, [7]. In this framework, we have the simulated environment of the North tower's 5th floor (IST building) and a simulated depth-color camera, that will provide the necessary data to calibrate a RGB camera. Having the RGB camera calibrated, methods developed to perform height estimates will be tested and evaluated. For the real environment, it is used an Asus X-tion(RGBD), that will provide 3D information to calibrate an Axis P1347 IP, and a mobile phone camera. These experiments were carried out in the North tower's 7th floor.

5.1 Calibration Results

Using now the Automated Calibration method, in order to evaluate their performance, in the VRML world [7], we consider the following experiments. In VRML world, a RGBD camera will obtain the depth information from the intersection of back-projection rays with the closest plane, simulating a real kinectic and providing the necessary 3D information for camera calibration. In VRML, the extrinsic and intrinsic parameters of uncalibrated camera are defined by the position, orientation, and field of view (FOV) of the viewpoint, so that we have a ground truth to compare with the obtained results.

So, we define the position and orientation of the RGBD and the RGB camera to test calibration method, see Fig. 5.1(a), and their respective views, Fig. 5.1(b) for RGB and Fig. 5.1(c) for RGBD, considering this last one as the depth data, see Fig. 5.1(d), required for the calibration.



(a) Setup (b) RGB image

(c) depth-color image (d) depth map

Figure 5.1: Camera calibration setup used in VRML (a), RGB image in (b), depth-color image in (c), and respective depth map in (d).

We start by obtain SIFT matches, as it can be seen in Fig. 5.2. These results will provide the data (M_i, m_i), to solve DLT-Points based approach, that will be used to estimate the 1st projection matrix approximation. Since the SIFT algorithm gives some bad matches (i.e. outliers), in order to reject them, a RANSAC based approach is used. Selecting six points from the SIFT results, it is performed a camera projection matrix estimate, and the number of inliers from all SIFT matches is counted (an inlier is defined in this case as being one pair (M, m) where the reprojection error is bellow a certain threshold, e.g. 1 pix). Reprojection error is defined as $Err = \sqrt{\sum(m - \hat{m})^2}$, where $\hat{m}$ is the estimate projection of M. The process is repeated until a minimum number of inliers is achieved. Having those inliers, the projection matrix is re estimated, see the green camera in Fig.5.4, where the outliers from SIFT were excluded. This estimate is used to match detected lines from Fig. 5.1(c), converted to 3D through Fig. 5.1(d), with lines from Fig. 5.1(b) (see Fig. 5.3 for fitted lines that were reprojected).

Adding these new fitted results constraints, i.e. the results from Fig. 5.3 to the previous one from SIFT Fig. 5.2, allows us to improve the camera calibration results, shown as the magenta camera in Fig. 5.4.

Knowing the position of the expected RGB camera, the ground truth, it is possible to estimate the position error from the performed calibration. Having a projection matrix P, it can be decomposed into $K[R\ t]$, B. With first estimate, green camera, it was obtained a position error

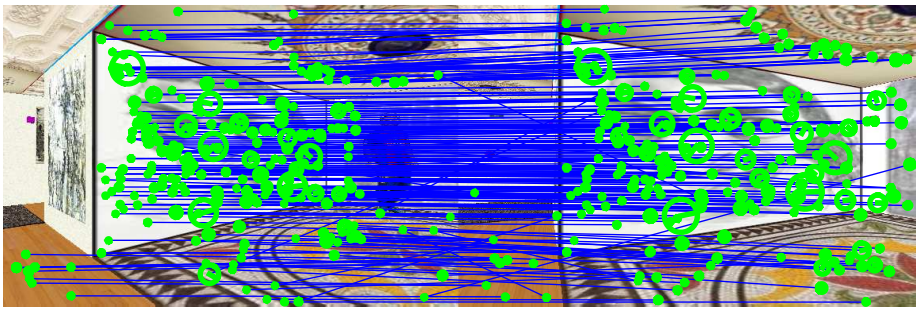


Figure 5.2: SIFT results obtained between RGB and depth-color image.

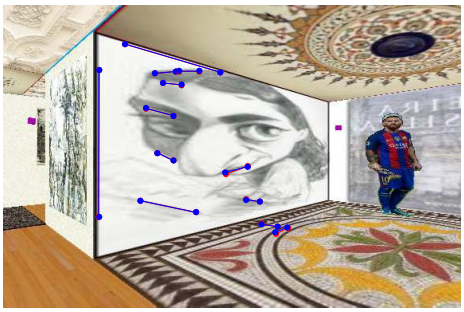


Figure 5.3: Lines re-projected and fitted using first projection matrix estimate. In red are simple reprojected lines while in blue are lines fitted with image gradient.

of 0.0105m, with SIFT. After adding the matching lines to this estimate, the results improved obtaining a position error of 0.0088m. The distance between rotation matrices was of 0.02 [rad] (this can be estimated using A), showing that fitted lines improve camera calibration estimation.

5.2 Non-Vertical Height Measurement

In order to perform height measurement of non-vertical pose, the following tests were performed. First, we test the 2D pose computation, which allows us to obtain the position of 2D joints in the RGB image, see the results in Fig.5.5. Using then these 2D RGB points and knowing camera intrinsic parameters, it is possible to use the method described in Sec. 4.2, to match these image points with the best 3D global pose.

Having now a 3D model, it is possible to estimate the depth of each joint using (4.15), and then perform one height estimate, by adding the segments as it is shown in Fig. 4.4. Having a picture of a person in same depth conditions and with the same resolution, so that the facet in VRML will have same height for both cases,sitting and standing, see Fig. 5.5(a) and Fig. 5.5(b). Since the facet where this person was declared (in VRML) has 1.6m of height, the estimated

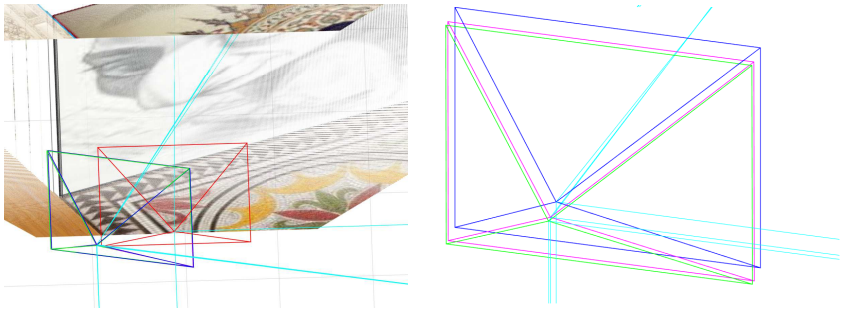
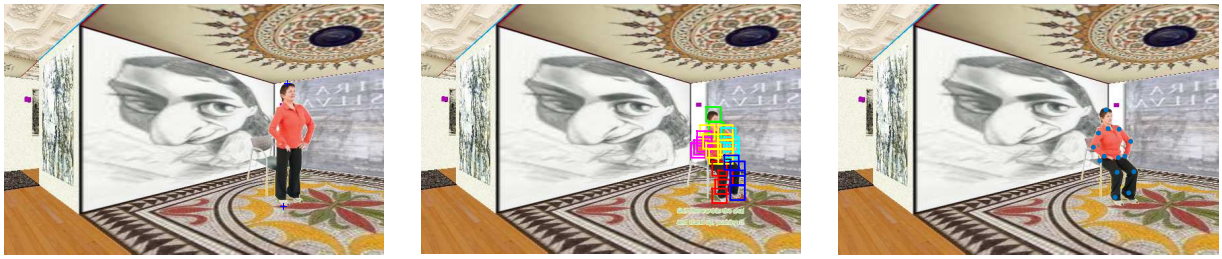


Figure 5.4: Calibration results obtained, RGBD camera in red, ground truth in blue, 1st estimate in green, 2nd improved estimate with lines in magenta.



(a) Person in vertical pose (b) Members position estimate (c) Joints position estimate

Figure 5.5: 2D person in vertical pose for comparison height estimation in (a), followed by person in non-vertical position detection in (b), that will lead to joints location estimation in (c)

height can be seen in table 5.1.

In order to test the height measuring for Non-Vertical Poses, at different depths, in a realistic scenario, it was used the game development platform called Unity [19]. In this scenario, with a pre-calibrated camera obtained from DLT-Points, it was taken four pictures of a character in a non-vertical pose, defined with a height of 1.84m, at different depths (see Fig. 5.7(a)). The estimated heights can be seen in the table 5.2.

As it can be seen in the predicted 2D joint position 5.7(b), the head is detected a bit higher than where it should be, which causes the estimated heights (see table 5.2) to have a slighty higher error. On the other hand, the obtained 3D model is pretty close to the character pose, Fig. 5.7(c).

5.3 Height Measurement Uncertainty Study

To analyze the height estimation uncertainty, a Monte Carlo methodology was used. In this method, repeated tests are performed in order to obtain numerical results, i.e. statistic data.

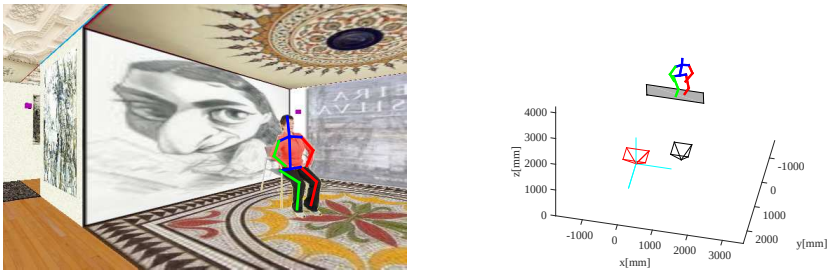


Figure 5.6: 3D Pose estimate with aligned camera in black and camera who took the picture in red.

Table 5.1: Height Estimate results VRML

True[m]	Standing[m]	Sitting[m]
1.6	1.6472	1.5370

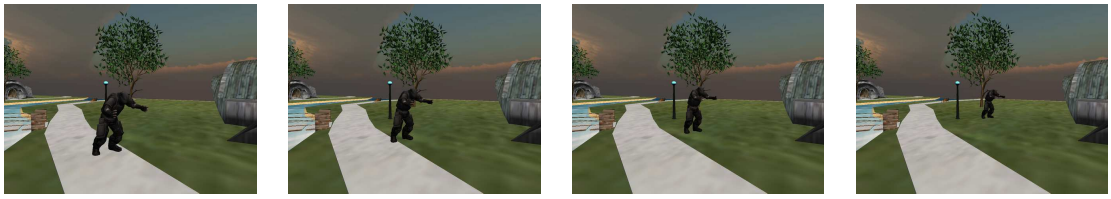
In different tests, it is added random noise to the user clicked point, and in some cases to the calibration data, in order to see how the described methods perform in presence of noise.

5.3.1 Noisy data SIFT and noisy SIFT based calibration

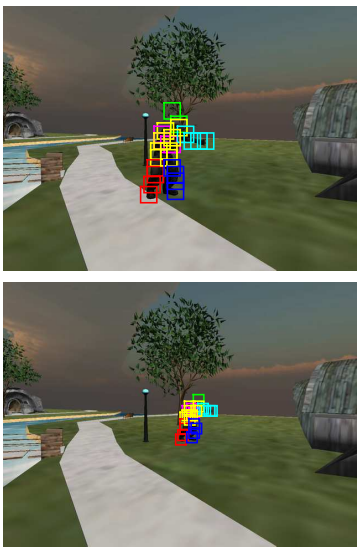
First, the use of the Automated Calibration without any noise allow us to perform a first height estimate, of a character defined in VRML with 1.7m. In this section, we apply a uniformly distributed noise to the points used to obtain that height, and performing 100 tests (Fig. 5.8) to estimate several heights with the some uncertainty, as it can be seen in Fig. 5.8(b) and Fig. 5.8(c). As expected, with the increase of the user uncertainty clicked points, the range of estimated heights will also increase. When noise is also added to calibration data, Fig. 5.8(c), the uncertainty increases but not significantly when compared to Fig. 5.8(b).

5.3.2 Height estimation vs depth

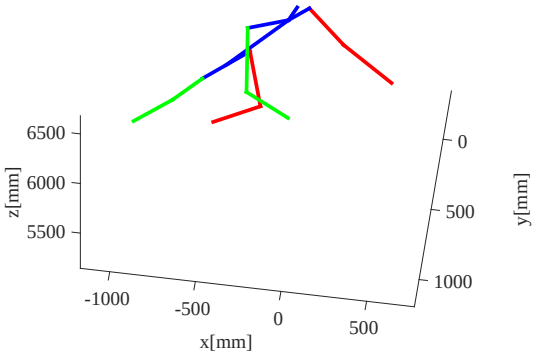
To analyse how the height estimation varies due to the depth and the camera tilting, 200 estimates were performed where the noise in user click is kept with constant value of 2 pix of standard deviation. We started these tests by changing the depth of the person in the scenario, as in Fig. 5.9(a) and Fig. 5.9(b). After several heights estimates, the results in Fig. 5.9(c) and in Fig. 5.9(d) show us that, with the increase of the depth, the uncertainty of the estimated height increases, Fig. 5.9(c). In addition, when the person comes closer to the camera, there is less uncertainty in height estimation, Fig. 5.9(d).



(a) Character at different depths in scenario



(b) Predicted 2D joints position



(c) 3D model found

Figure 5.7: Different depth of character in scenario in (a), person detection results in (b) for different depths, 3D model predicted (c) from 2D detection in one of the cases.

5.3.3 Height estimation vs tilting

Next, we change the tilting in camera, as it can be seen in Fig. 5.10(a), from $0°$ to $30°$, and performing 200 height estimates with the same 2 *pix* standard deviation of noise in user click. The results are shown in Fig. 5.10(b). It can be seen that, when estimation is near one of the corners of the picture, the uncertainty is smaller Fig. 5.10(c).

5.4 Real-World Datasets

In this section, the experiments using real data are described. It is used an Asus X-tion(RGBD), that will provide 3D information to calibrate an Axis P1347 IP. In the first experience, using the Automated Calibration method described in 3.4, we obtain a camera calibration that will be used to perform some measures. For the next experience, it is used the same 3D data provided

Table 5.2: Height Estimate results in Unity

True[m]	Increasing Depth			
1.84	1.9192	1.8546	1.8754	1.9865

by the Asus X-tion(RGBD), but now to calibrate a mobile phone camera, using the Assisted Matching of Lines method, from Sec. 3.3, and the previously described method to make some height estimates.

5.4.1 Measures with Automated Calibration

Now, it is performed a test for extracting measures using real data, in order to evaluate the described method. In this experiment, the objective is to estimate the height of an "air conditioning" in the 7th floor of IST's north tower, using a RGB image of an uncalibrated camera (Fig. 5.11(a)) and depth-color data from a kinetic camera, Fig. 5.11(b) and (c).

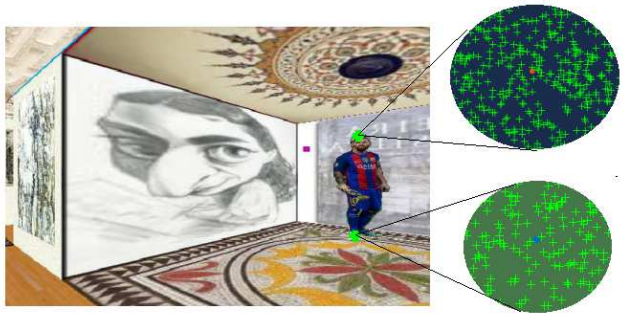
Applying the method described in image Fig. 5.11(b), on a cropped image Fig. 5.11(a), it was performed a 1st estimation of projection matrix of camera, as it can be seen in Fig. 5.11(d). Then, using this estimate to match lines between RGB and depth-color image, it was possible to improve the projection matrix estimate.

The obtained distance between RGB and RGBD cameras from the performed calibration was 3.26m. Since we know that the distance between them is 3.55m, one can conclude that it was possible to estimate the RGB position with an error of 29cm. Then, we use the obtained projection matrix to estimate the height of the air conditioning, as it can be seen in Fig. 5.12.

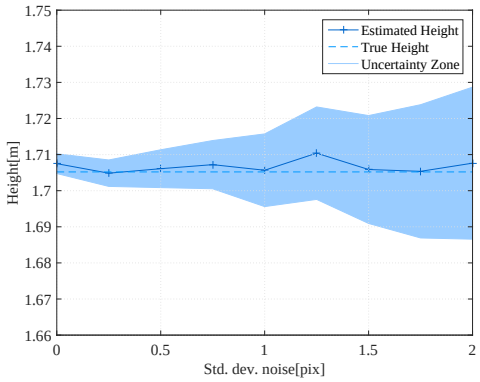
Applying a 0.5pix deviation to these clicks allows us to conclude that the closest height estimated value was of 66cm. Since we know that the air conditioning height is about 63cm, we can conclude that the height was estimated with an error of 2.4cm.

5.4.2 Person height measure with Assisted Matching of Lines

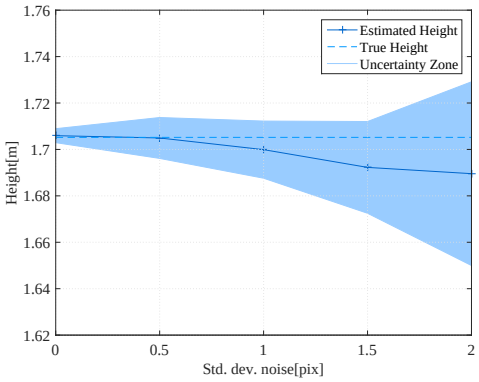
Now using a depth-color image and a depth image as in the previous example, Fig. 5.11(b) and (c), but using an RGB image of a mobile phone, Fig. 5.14(a), our goal is to estimate the height of a subject. Since the SIFT algorithm did not provide good enough results, it was used the Assisted Matching of Lines method, where it is asked for a user to perform the matching of lines, and pick some more while line fitting techniques are being used. The data used for camera calibration can be seen in Fig. 5.14(a) and (b). The estimated camera can be seen in point cloud of Fig. 5.14c).



(a) Uniformly distributed noise applied to user clicked (user point in orange, and blue, noisy points in green).

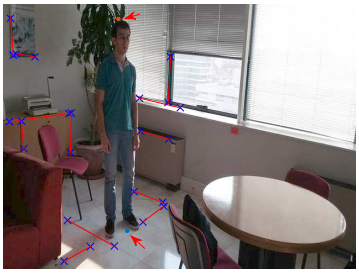


(b) Noisy click SIFT based calibration

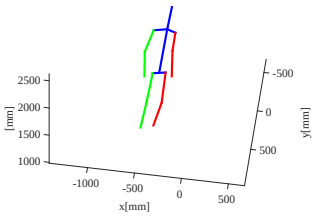


(c) Noisy click and noisy data SIFT based calibration

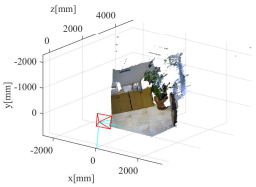
Figure 5.8: Uniformly distributed noise applied to user clicked (user point in orange, noisy points in green) in (b) and noise applied to user click and calibration data in (c).



(a) RGB with calibration data



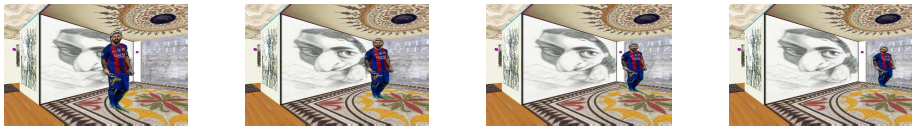
(b) 3D model



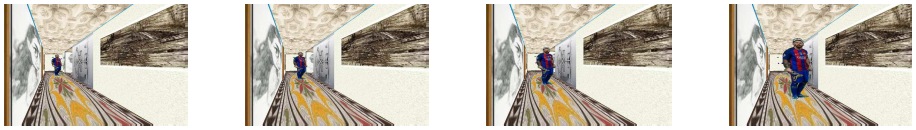
(c) PointCloud

Figure 5.13: Height Estimation of person standing using RGB image obtained with mobile phone camera and calibration data from Assited Matching Lines method (a).3D predicted model from person in image in (b).Results showing estimated mobile phone camera position over the pointcloud (c).

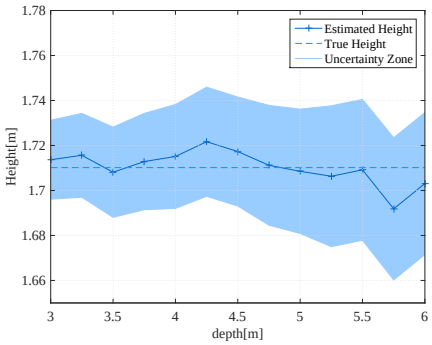
The methods described previously in Sec. 4.2 are used for 3D pose model estimation, that



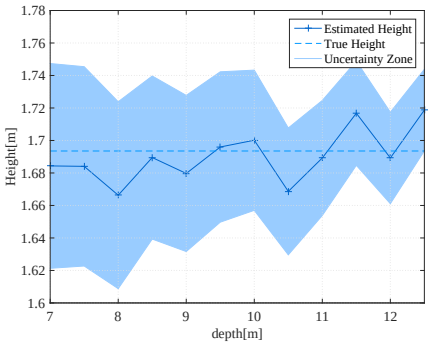
(a) Different depth in hall



(b) Different depth in corridor



(c) Height mean and std dev. vs depth in hall



(d) Height mean and std dev. vs depth in corridor

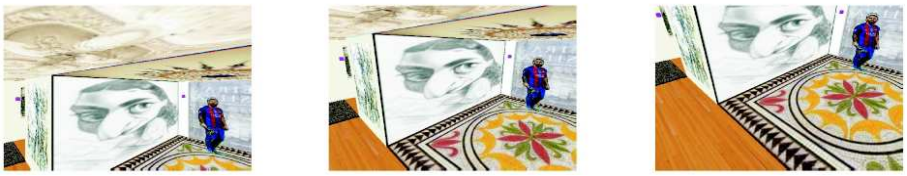
Figure 5.9: Different depth in hall starting at 3m from origin frame on the left wall going up to 6m (a), with respective height mean and std dev in (c), different depth in corridor starting at 7m and going to 12.5m (b), with respective height mean and std dev in (d).

will provide an height estimate to be compared with the individual true value of height, 1.84m.

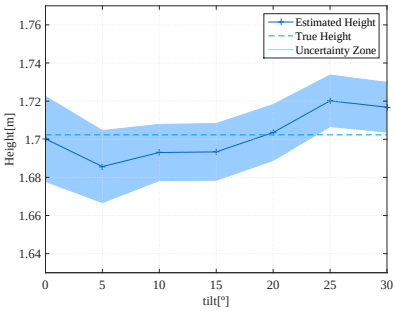
Now, we use the methods developed for vertical height estimate: the one where it is used the click on head and on feet; and the one who uses predicted model. Results are presented in the table 5.3, for both methods. User clicked points can be seen in Fig. 5.13(a).

As it can be seen from Fig. 5.14(d) and (e), the predicted 2D joint have slight error, left arm of subject is lifted and hip points considered are higher than it should be, which make the height estimate (with the predicted model) to have a slightly higher error. When those points are adjusted by the user, the *"Aided Predicted Model"*, Fig. 5.14(f), pose improves making therefore the height estimate more accurate.

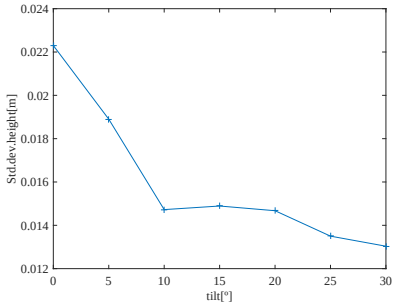
As it can be seen, both methods present some errors. These errors might came from the camera calibration itself,since it will propagate the errors trought distance measures to height estimate.



(a) Various camera tilt angles



(b) Height mean and std dev. vs tilting



(c) Std dev. vs. tilting

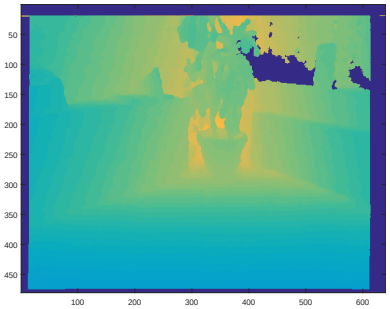
Figure 5.10: Different tilting as it can be seen in (a) from $0°$ to $30°$, respective estimated heights in (b) and their uncertainty in (c).

Table 5.3: Height Estimate Results Real Data

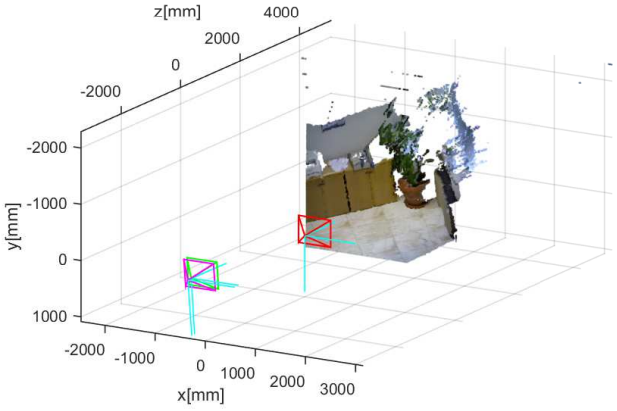
Pose	Method	Height Estimate [m]	Err[%]
Vertical	Click method	1.8008	2.13%
	Aided Model	1.7582	4.45%
Non-Vertical	Predicted Model	1.7135	6.87%
	Aided Predicted Model	1.8164	1.28%

(a) Image of the camera to calibrate

(b) RGBD data



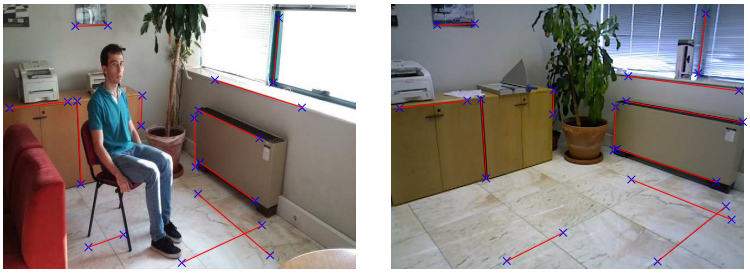
(c) RGBD depth map



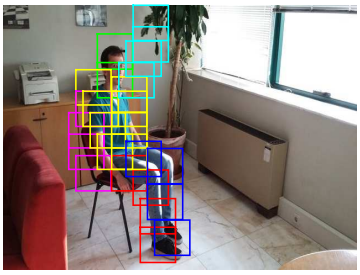
(d) RGBD camera location and
two estimated locations of the RGB camera

Figure 5.11: Calibration of a fixed surveillance camera using a mobile color-depth (RGBD) camera. (a) RGB image of calibrated camera to calibrate. (b) and (c) show color and depth images acquired by a Microsoft Kinect. (d) Results showing the location of the color-depth camera (red) and estimates of the locations of surveillance camera, 1st estimate in green and 2nd estimate in magenta, all drawn over the point cloud acquired by the color-depth camera.

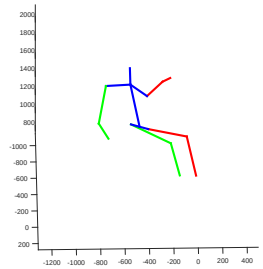
Figure 5.12: Clicked points by user in blue and orange to perform vertical height estimate. On the right value of air conditioning height.



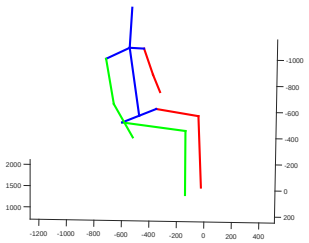
(a) RGB (b) RGBD (c) PointCloud



(d) 2D pose estimate



(e) 3D model



(f) 3D model using user aid in hip click

Figure 5.14: Height Estimation of person sitting using RGB image obtained with mobile phone camera and calibration data from Assited Matching Lines method (a).Depth-color image obtained with Microsoft Kinectic (b).Results showing estimated mobile phone camera position over the pointcloud (c). 2D person members detection in (d). 3D predicted model in (e) from results in (d). 3D model predicted with 2D joints positions aided by the user(f).

Chapter 6

Conclusion and Future Work

The main objective of this project was to perform the estimate of the height of a person captured in image of a mobile phone camera. In order to calibrate such mobile phone camera and just like in most forensic investigations it is possible to visit again the scenario and use measurement equipment such as laser range finders to obtain 3D data of the scenario required for calibration.

In particular the methods used for camera calibration were studied, and how to perform a good calibration of camera. It was also studied, given a mobile phone camera photo how to detect person and perform estimate of his height for different poses.

The methods developed for camera calibration try to improve the ones developed previously, by automating them with tools such as SIFT algorithm. This method allowed to obtain reasonable results in camera calibration when SIFT provided good matches results. If not some user input would help in calibration.

This work presented a method for estimating the height of a person standing in various different poses. It uses pose estimate based on matching of detected person position in image with dataset of different human 3D poses, if the person detection in image has some bad results these can also be aided by user input allowing to improve the matching with 3D dataset pose. After having proper human pose, it is possible to perform estimate of human height.

The uncertainty in height associated with variables such as the tilting of camera or the depth of the person in the scenario is analyzed in order to obtain the best installation parameters of surveillance cameras in certain scenarios.

In future work we determine the relationship between random perturbations in the input data (noise) and the calibration errors for the proposed methodology of height estimation in non-vertical case. These rules will serve the purpose of building user interfaces helping the height estimation in captured images from surveillance cameras.

Appendix A

Estimate Angle between two Rotation Matrices

A.1 Rotation matrix properties

A rotation matrix transforms the set of coordinates representing a three dimensional object object, in an orthogonal Cartesian frame, without changing its shape or size, i.e. the length of any vector and the angle between any pair of vectors are unchanged. Such a matrix is called orthonormal and has several properties which follow from this definition.

A.1.1 The columns of a rotation matrix are orthogonal unit vectors

A rotation matrix may transform any set of vectors, so we can consider transforming the three unit vectors along the x,y and z axes, which by definition are orthogonal to each other

$$R(x\ y\ z) = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \text{(r1 r2 r3)}.$$

Since the starting vectors x,y,z are orthogonal and of unit length, then the resultant vectors r1,r2,r3 must be also, these are the columns of the rotation matrix R.

A.1.2 The transpose of a rotation matrix is its inverse

Multiplying the rotation matrix R by its transpose $\mathbf{R}^T$ (in which the columns of R become the rows of $\mathbf{R}^T$),

$$\mathbf{R}^T\mathbf{R} = \begin{bmatrix} r_1^T \\ r_2^T \\ r_3^T \end{bmatrix} \begin{bmatrix} r_1 r_2 r_3 \end{bmatrix} \tag{A.1}$$

Since r1, r2 and r3 are orthogonal unit vectors, then $r1 \cdot r1 = 1$,$r1 \cdot r2 = 0$ etc.

$$\mathbf{R}^T\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

hence $R^T = R^{-1}$, since this is the definition of an inverse matrix R^{-1}.

A.1.3 Geometric aspects of the exponential and logarithm

For any vector $u = (u_1, u_2, u_3) \in R^3$ one can associate 3x3 skew-symmetric matrix.

$$S_u = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}$$

It is easy to see that this matrix simplifies the cross product between two vectors $u \times v = S_u v$, So considering the following problem: given a unit vector $u \in R^3$ and an angle θ, find the rotation matrix R that rotates any vector through the angle θ about an axis with direction u. The solution is the matrix exponential.

$$R = e^{S_u \theta} \tag{A.3}$$

where the matrix exponential e^X is defined as:

$$e^X = \sum_{k=0}^{+\infty} \frac{X^k}{k!} \tag{A.4}$$

The inverse problem, obtaining the angle θ and vector u, from a rotation matrix can be done by the matrix logarithm.

$$S_u \theta = log(R) \tag{A.5}$$

or equivalentily

$$S_u = \frac{1}{\theta} log(R) \tag{A.6}$$

When $\theta \neq 0$ and applying norms it yields

$$\|S_u\||\theta| = log(R) \tag{A.7}$$

and since $\|S_u\| = 1$ the angle can be obtained by

$$|\theta| = \|log(\theta)\| \tag{A.8}$$

A.2 Rotation between two matrices

For any two rotation matrices $R1, R2 \in R^3$ it is possible to use property from 2.1.2 where a $R^T = R^{-1}$, if we multiply

$$R_{12} = R1 \cdot R2^T \tag{A.9}$$

we obtain R_{12} which represents the difference angle between their rotations. It is then possible to use (A.8) that will allow us to obtain the rotation angle between them, and (A.6) to obtain the vector of rotation.

Appendix B

Projection Matrix Decomposition

Projection matrix P, from $m = PM, P = K[Rt]$ can be decomposed in the intrinsic parameters matrix K, a rotation R and a translation t. The notation $P_{a:b,c:d}$ denotes the selection of lines (a to b) or columns (c to d). The intrinsic parameters matrix, K is assumed to have an upper triangular form:

$$K = \begin{bmatrix} f_u & s_k & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{B.1}$$

where (f_u, f_v) represent scaling from meters to pixel coordinates, s_k is the skew coefficient, and $(c_u, c_v)^T$ represents the coordinates of principal point. Since the rotation matrix is unitary and thus combined with K implies that $\|P_{3,1:3}\| = 1$

Given a projection matrix P, we want to decompose it in its intrinsic and extrinsic parameters. For that the following steps are used: (i) QR factorization of P, (ii)transformation from QR to RQ factorization, and (iii) sign correction of K.

B.1 QR based on Gram-Schmidt orthonormalization

This factorization can be obtained as the unitary matrix the orthonormal vectors found from the matrix to be factorized:

$$P_{1:3,1:3} = Q \cdot R = [q1\ q2\ q3] \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{B.2}$$

where q_i denotes the orthonormalized vectors of $P_{1:3,1:3}$, for instance $q_1 = P_{1:3,1}/\|P_{1:3,1}\|$, and $r_{i,j}$ are weighting factors found in the orthonormalization process.

B.2 Converting QR to RQ

The QR decomposition factorizes $P_{1:3,1:3}$ as a unique product of an orthonormal (unity) matrix and an upper-triangular matrix, provided that the upper-triangular has the main diagonal entries all positive. Note that we want the reverse, i.e. an upper-triangular times an orthonormal, as in the projection equation (2.6). We will see next that the QR factorization can be easily transformed to RQ. Defining a matrix S as:

$$S = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{B.3}$$

S has the property that left(right) multiplying 3x3 matrix swaps its lines (columns). In addition $S^T = S$ and $S \cdot S = I$. The applying the QR factorization to $P_{1:3,1:3} \cdot S$

$$P^T_{1:3,1:3} = Q \cdot U \tag{B.4}$$

and doing some algebraic operations starting with a transpose and then using the properties of S, one has $P_{1:3,1:3} = S \cdot U^T \cdot Q^T = S \cdot U^T \cdot S \cdot S \cdot Q^T = (SU^TS) \cdot (SQ) = K \cdot R$ this is already a RQ factorization as $K = SU^T$S comprises the necessary line and column swapping to change the lower-left-triangular U^T to upper-triangular, and the $R = S \cdot Q^T$ is still unity as the transpose and the operation of S do not affect the unity property.

B.3 Correcting the diagonal K

The QR and RQ factorizations can leave a sign ambiguity, which is removed by requiring that K has positive diagonal entries. Defining a diagonal matrix D having the signs of the main diagonal of K,

$$D = diag\left\{sign(K_{1,1}), sign(K_{2,2}), sign(K_{3,3})\right\} \tag{B.5}$$

one can correct the signs of K, and consequently update all the terms of the factorization with:

$$K \leftarrow KD, R \leftarrow DR, t \leftarrow= K^{-1}P_{1:3,4} \tag{B.6}$$

The algorithm of decomposing a matrix using matlab QR factorization is the following:

$[K, R, t] = \texttt{proj\_decomp}(P)$
% RQ from QR factorization
$S = [0\ 0\ 1; 0\ 1\ 0; 1\ 0\ 0];$
$[Q, U] = qr(P(1:3, 1:3)^{'} * S); K = S * U^{'} * S; R = S * Q^{'};$
% Correcting signs and computing t
$D = diag(sign(diag(K)));$
$K = K * D; R = D * R; t = inv(K) * P(:, 4);$

Bibliography

[1] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1014–1021, June 2009.

[2] A. Aziz and H. Karara. Direct linear transformation into object space coordinates in close-range photogrametry. In *Proc. of the Symposium on Close-Range Photogrammetry.*, pages 1–18, 1971.

[3] M. Berhanu. *The Polynomial Eigenvalue Problem*. Ph.d. thesis, University of Manchester-School of Mathematics, 2005.

[4] Jean-Yves Bouguet. Camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj`.

[5] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, 4(2):127–139, Mar 1990.

[6] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation = 2d pose estimation + matching. *CoRR*, abs/1612.06524, 2016.

[7] W3 Consortium. Virtual reality modeling language. `http://www.w3.org/MarkUp/VRML/`.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005.

[9] F. Endres, J. Hess, N. Engelhard, J. Sturm, and W. Burgard. Openslam. `http://openslam.org/rgbdslam.html`. Accessed in 2012-10-22.

[10] A. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition.*, volume 1, pages 125–132, 2001.

[11] Ricardo Galego, Agustin Ortega, Ricardo Ferreira, Alexandre Bernardino, Juan Andrade-Cetto, and JosÃ© Gaspar. Uncertainty analysis of the dlt-lines calibration algorithm for cameras with radial distortion. *Computer Vision and Image Understanding*, 140:115 – 126, 2015.

[12] Michael Grossberg and Shree K. Nayar. *The Raxel Imaging Model and Ray-Based Calibration*, 02 2005.

[13] M. Hansard, R. Horaud, M. Amat, and S. Lee. Projective alignment of range and parallax data. In *Proc. IEEE Conf. Comp. Vision and Pattern Recognition.*, pages 3089–3096, 2011.

[14] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[15] R. I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, Jun 1997.

[16] R. I. Hartley. Kruppa's equations derived from the fundamental matrix. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):133–135, Feb 1997.

[17] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1106–1112, Jun 1997.

[18] V. Ila, J. Andrade-Cetto, R. Valencia, and A. Sanfeliu. Vision-based loop closing for delayed state robot mapping. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3892–3897, Oct 2007.

[19] Simon Jackson. *Unity 3D UI Essentials*. Packt Publishing, 2015.

[20] T. Kazik, L. Kneip, J. Nikolic, M. Pollefeys, and R. Siegwart. Real-time 6d stereo visual odometry with non-overlapping fields of view. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1529–1536, June 2012.

[21] Istvan Kispal. *HUMAN HEIGHT ESTIMATION USING A CALIBRATED CAMERA*, 01 2008.

[22] L. Kneip and H. Li. Efficient computation of relative pose for multi-camera systems. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 446–453, June 2014.

[23] N. Leite. Calibração de uma rede de câmaras baseada em odometria visual. Master's thesis, UTL - Instituto Superior Técnico, 2009.

[24] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[25] H. LÃ¼tkepohl. *Handbook of Matrices*. Wiley and Sons, 1996.

[26] D. Marr and H. K. Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London B: Biological Sciences*, 200(1140):269–294, 1978.

[27] M. Silva, R. Ferreira, and J. Gaspar. Camera calibration using a color-depth camera: Points and lines based dlt including radial distortion. In *Workshop in Color-Depth Camera Fusion in Robotics, IEEE IROS*, 2012.

[28] Manuel Silva, Ricardo Ferreira, and JosÃ© Gaspar. *Camera Calibration using a Color-Depth Camera: Points and Lines Based DLT including Radial Distortion*, 08 2017.

[29] R. Tsai. A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras. *IEEE J. Robot. Automat.*, 3(4):323–344, Aug 1987.

[30] Yue Wang. *Non-contact Human Body Measuring Technology Based on Camera Calibration Technique*, 12 2015.

[31] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR 2011*, pages 1385–1392, June 2011.

[32] Zhengyou Zhang. A flexible new technique for camera calibration, 2002.