Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Construction of Laser Plane Rangefinder (LPRF)

Jakub Cmíral

Supervisor: Ing. Pavel Krsek, Ph.D.
May 2017

Acknowledgements

I am grateful to my supervisor Dr. Pavel Krsek for guiding me and enabling me to comprehend a little what is research. I appreciate help of other members of the Robotic Perception Group at CIIRC ČVUT, namely prof. Václav Hlaváč, Vladimír Petrík, and Dr. Martin Matoušek.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instruction for observing the ethical principles in the preparation of university theses.

Prague, May 22, 2017

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl že jsem uvedl veškeré informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolské závěrčné práce.

V Praze, 22. května 2017

Abstract

The software tool supporting a implementation of a cheap laser plane range finder (LPRF) with rotary table and its calibration is presented. We were motivated by a dual-arm robotic manipulation with soft free-form objects, e.g. a piece of garment. We need a precise surface measurement also as a reference data for the precision estimation of other depth acquiring methods as stereo vision.

A laser diode with a cylindrical lens generates a light plane observed by a camera in LPRF. The distance to scene points is obtained by triangulation. As LPRF construction depends on a particular class of objects size and their geometry, the device has to be often built for a new application anew.

We prepared a software tool in Python, intended for the public domain, which aids and simplifies the calibration and precision evaluation for such new LPRF constructions. The thesis describes the functionality, calibration procedure, precision evaluation methodology and the implementation. The novelty and gain for the reader are in simplicity and easy use for such a rather frequent application.

Keywords: 3D reconstruction, camera calibration, depth sensor, LPRF

Supervisor: Ing. Pavel Krsek, Ph.D.
Czech Technical University in Prague,
Czech Institute of Informatics, Robotics and Cybernetics
Jugoslávských partyzánů 1580/3,
Prague 6, 166 36,
Czech Republic

Abstrakt

Implementace a konstrukce laserového hloubkového snímače (LPRF) s rotačním stolkem společně s jeho kalibrací. Byli jsme motivováni dvourukou robotickou manipulací s měkkými předměty bez předem daného tvaru (kus látky). Potřebovali jsme přesná a zároveň referenční data pro jiné metody hloubkové 3D rekonstrukce, např. stereo vidění.

Laserová dioda s cylindrickou čočkou emituje laserovou rovinu a vytváří laserovou stopu pozorovanou kamerou. Vzdálenost kamery od laserové stopy je počítána triangulací mezi kamerou a laserovou stopou. Konstrukce LPRF záleží na velikosti skenvaného objektu a jeho geometrii a je potřeba ji pro každou aplikaci mněnit.

Připravili jsme softwarový nástroj v Pythonu, který ulehčuje kalibraci a dokáže určit přesnost LPRF v jeho nové konfiguraci. V práci je popsána funkčnost, kalibrační proces, určení přestnosti zařízení a jeho softwarová implementace. Přínosy pro čtenáře jsou v jednoduchosti and snadném použití této velmi běžně používané věci.

Klíčová slova: 3D rekontrukce, kalibrace kamery, hloubkový sensor, LPRF

Překlad názvu: Konstrukce laserového hloubkového snímače (LPRF)

Contents

Figures

Tables

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Jakub C m í r a l

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Bachelor Project: Construction of Laser Plane Rangefinder (LPRF)

Guidelines:

1. Study principle of laser plane rangefinder (LPRF), existing similar devices and related software.
2. Construct the LPRF for measuring objects (approximate dimensions 25x25x15 cm).
3. Design calibration process and create a program for calibration of the LPRF.
4. Create a program for measuring of objects on LPRF.
5. Implement software as universal library, which can be used for LPRF with modified geometry (different dimensions and arrangement of the components).
6. Prepare detail documentation to allow using library by third party.

Bibliography/Sources:

[1] M. Sonka, V. Hlavac, R. Boyle: Image Processing, Analysis and Machine Vision. Thomson, 3rd edition, ISBN 978-0-495-08252, 2007.
[2] R. Hartley and A. Zisserman: Multiple view geometry in computer vision. Cambridge University, 2nd edition, ISBN 0-521-54051-8, 2003.
[3] P. F. Sturm and S. J. Maybank: On plane-based camera calibration: A general algorithm, singularities, applications. In CVPR, ISSN: 1063-6919, pages 1432-1437. IEEE Computer Society, 1999.
[4] Z. Zhang: Flexible camera calibration by viewing a plane from unknown orientations. In Proc. Int. Conf. Computer Vision (ICCV), ISBN: 0-7695-0164-8, pages 666-673, IEEE, 1999.

Bachelor Project Supervisor: Ing. Pavel Krsek, Ph.D.

Valid until: the end of the summer semester of academic year 2017/2018

L.S.

prof. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 9, 2017

Chapter 1

Introduction

My work stems from the CloPeMa (Clothes Perception and Manipulation) project legacy. There has been a need to work with general free form surfaces of a general piece of fabric, e.g. a wrinkled towel. Such an (outer) surface constitutes 2D manifold in 3D space. The visible part of the manifold can be sensed as a depth map by the range finder. We needed a rather precise capturing device measuring a reference depth map. It serves for evaluating fabric understanding methods, which often use less accurate depth maps, e.g. of CloPeMa testbed range finders (Kinect1-like and stereo vision).

We built a rather inexpensive laboratory version of the laser plane range finder (LPRF). Parts of LPRF are a laser diode with a cylindrical lens projecting a laser plane, a digital camera with interchangeable lens, and a computer controlled rotary table, which is the most expensive piece of the setup. The measured object is placed on the rotary table. The laser plane illuminates the laser trace on the surface of the object creating its cut. The depth map is obtained by observing a projected laser plane by a camera. The observed bright red line stemming from the projected light plane is observed by a camera. The depth is calculated by triangulation. The projected laser plane allows finding correspondences.

LPRF should be reconfigurable for other applications when measured objects have, e.g. different size. The calibration of the rangefinder is needed to get metric measurements. The calibration has to be performed repeatedly in practice because the setup configuration changes or rangefinder pieces were moved mechanically, e.g. because of temperature changes, device movement, vibrations, etc.

Such LPRFs have been used widely both in academia and industry since 1990s. Our team[1] has had an experience with it. Nevertheless, we did not have a handy piece of code for the purpose. The code for LPRF with rotating table found in public domain was hard to configure for the changed setup. Most of them work with specific hardware. The need to build and use such LPRF is common. Besides solving our particular assignment of measuring wrinkled towel-like objects, we desired to document the method and to create a public domain software tool for this purpose.

[1]The Robotic Perception Group from CIIRC ČVUT Prague.

1.1 Principle description

LPRF basic principle resembles stereo vision [18]. One camera of the stereo camera pair is replaced in LPRF by a laser projector, which creates a light plane and illuminates the measured object. The light plane creates the broken straight 'light trace' in the image resembling a cut of the object. The light trace is easily detectable in the camera image. The epipolar constraint reduces the search for corresponding points space to 1D similarly to stereo vision. The light trace provides only one distinct point on the epipolar line, which simplifies the correspondence problem. As both laser plane source and the observing camera are in certain distance each from the other (called a baseline), the depth is calculated by triangulation for all points in correspondence. More cuts are needed to measure the whole object. There are translational LPRFs, hand-held free movement LPRFs and our chosen LPRFs with a rotary table. Basic principle is shown in Figure 1.1, where (1) is the camera, (2) is the laser plane emitter, and (3) is the scanned object.

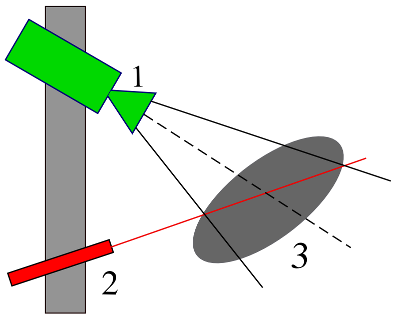


Figure 1.1: The basic principle of LPRF.

1.2 Task formulation

The task has been to create a duplicable, and reconfigurable laboratory LPRF hardware and its modular software. There has been a need to:

1. Develop and implement method providing a depth map.
2. Develop and implement the calibration method providing the calibration parameters and the assessment of the measurement precision.
3. Test the developed methods.
4. Document the procedures above and put it into the public domain.

The growing popularity of a freely accessible Python language, its development environments and rich libraries motivated us to use them in the reported work.

Chapter 2

Related work of others

The laser plane range finder (LPRF) is one of the popular solution for the depth maps capturing. It is also known as a 3D scanner or a structured light scanner.

There are general purpose commercial 3D Vision libraries, which provide the alternative solution of our task. For example, HALCON library [13] supports laser scanning, camera calibration, 3D transformations, stereo vision, single camera measuring, etc. The library is costly and is not an open platform. There are other 3D vision libraries such as National Instruments LabVIEW 3D Machine Vision Library [1], which is similar to HALCON library.

There are hardware platforms with proprietary software. For example COGNEX 3D Displacement Sensor [4] (Figure 2.1a) provides more functionality than the 3D scanning. FARO presents Design ScanArm [7] (Figure 2.1b), which is a 3D scanning robot arm with a blue laser module. The arm handled by the operator and the arm proportions cannot be changed. It uses Geomagic software [9]. FARO solution is neither modular nor an open platform.

(a) : COGNEX 3D Displacement Sensor. Courtesy [4].

(b) : Faro Design ScanArm. Courtesy [7].

Figure 2.1: The hardware platforms with proprietary software.

3D scanner [11] (Figure 2.2), which was originally called DAVID and purchased by HP in the mid 2016, is a completely closed platform, a free to use software with a ready to use hardware. DAVID focuses primarily on photometric scanning. It supports the structured light scanning too.

We also look for some open-source libraries and projects. We found some

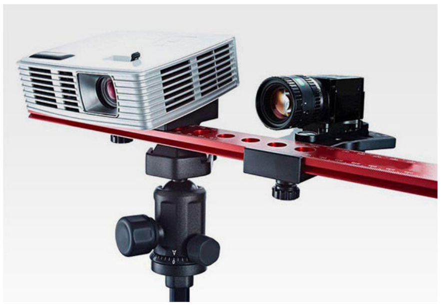
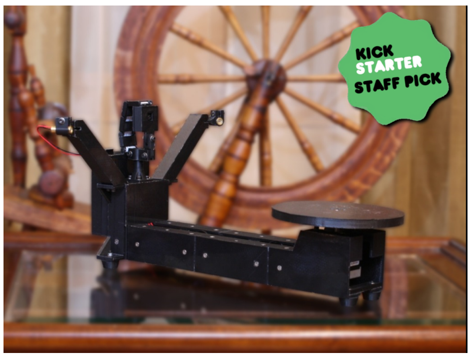


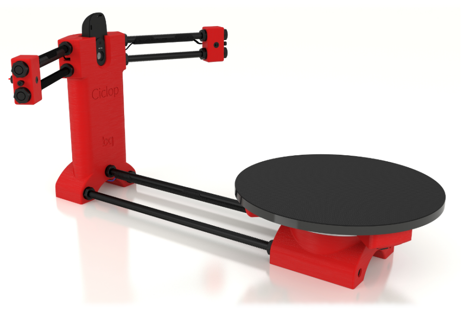
Figure 2.2: DAVID 3D scanner by HP. Courtesy [11].

projects such as FreeLss [8], Atlas 3D [12] (Figure 2.3a), BQ Ciclop, [3] (Figure 2.3b), and Horus [10]. Other projects existed, which are no longer supported or undocumented. Atlas 3D is laser plane scanner with two lasers, the camera and rotating table, GUI FreeLSS and a basic calibration. FreeLSS is undocumented.

Ciclop is a do-it-yourself 3D scanner accompanied with the software Horus. BQ Ciclop is a laser scanner with a rotating table similar to Atlas 3D. Ciclop itself is not interesting for us. Horus is open-source GUI and a 3D scanning library used with Ciclop. Horus is the multi-platform application for experiments with BQ Ciclop. Horus supports only Logitech C270 camera with a relatively low resolution (1280x960 px), a fixed focal length, and the depth of field approximately 300 mm. If there is a need to scan an object, which is out of focus, the camera optics has to be disassembled and refocused.



(a) : Atlas 3D. Courtesy [12].



(b) : BQ Ciclop. Courtesy [3].

Figure 2.3: Open source projects.

Kinect 2 [5] (Figure 2.4) is the other rather inexpensive 3D scanning device. Its limitations stem from a minimal scanning distance ≈ 50 cm, and relatively low precision ≈ 2 mm in its range of scanning distances [23]. Actually, our LPRF should provide a more accurate reference measurements applications than Kinect 2.

None of the above reviewed devices suits our purpose. These platforms either not open or are limited to a specific hardware. We decided to create an own LPRF software tool suited diverse hardware configurations, and

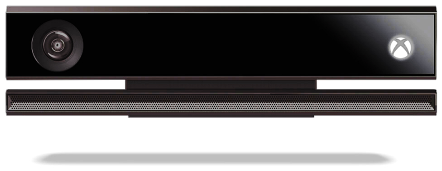


Figure 2.4: Microsoft's Kinect 2. Courtesy [5].

implement it as a Python package. The tool should be open, modular and configurable.

We considered initially to base our construction and implementation on 'in-house' LPRF [24]. This LPRF software was written in C++ and MATLAB for one specific device. However, we aimed at a new solution/implementation, a modular and open platform one.

Chapter 3
Proposed solution, theory, calibration

3.1 Camera model, model parameters calibration

A pinhole camera model is assumed. The camera model without a skew is used [18]

$$s\,\vec{u} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\mathbb{R}|\vec{t}\,\right] \vec{X} = \mathbb{K} \left[\mathbb{R}|\vec{t}\,\right] \vec{X}, \tag{3.1}$$

where $\vec{X} = (X, Y, Z, 1)$ are the global homogeneous coordinates of a point, $\vec{u} = (u, v, 1)$ are point coordinates in the image, (c_x, c_y) is a principal point, (f_x, f_y) is the focal length. $[\mathbb{R}|\vec{t}\,]$ is the matrix of extrinsic parameters. It is used to describe the position/orientation of the camera in a global coordinate system. Figure 3.1 illustrates the pinhole camera model.

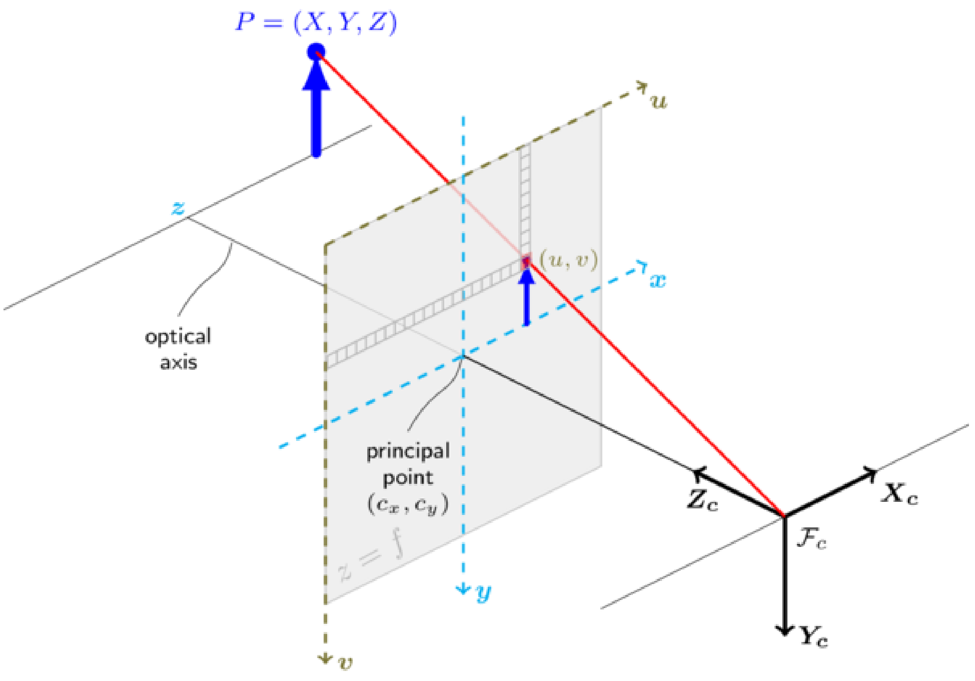


Figure 3.1: Pinhole camera model. Courtesy [15].

We use the pinhole camera model with radial and tangential distortion of the lens [15, 18]. The distortion model is

$$z \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = z \begin{bmatrix} x/z \\ y/z \\ 1 \end{bmatrix} = \mathbb{R} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \vec{t}, \tag{3.2}$$

$$x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2 p_1 x' y' + p_2 (r^2 + 2 x'^2) ,$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1 (r^2 + 2 y'^2) + 2 p_2 x' y' ,$$

$$u_{\text{corrected}} = f_x x'' + c_x ,$$

$$v_{\text{corrected}} = f_y y'' + c_y ,$$

where $r^2 = x'^2 + y'^2$, (k_1, k_2, k_3) are radial distortion coefficients and (p_1, p_2) are tangential distortion coefficients.

The camera calibration estimates internal, external calibration parameters and distortions coefficients. We use camera calibration implemented in OpenCV2 library, which is based on articles [19, 25]. A C++ implementation is also available in OpenCV2 together with its Python wrapper. The implementation supports several calibration patterns. We choose a flat black and white chessboard pattern for our calibration processes, see Figure 3.2.

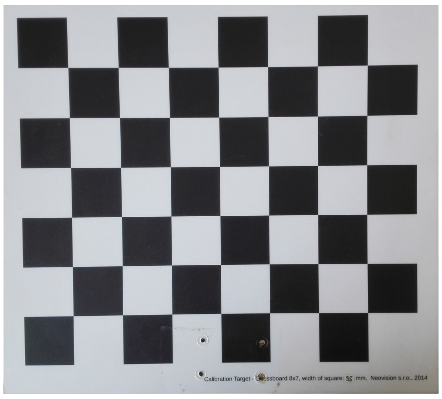


Figure 3.2: The calibration pattern 8x7 with the square size 35 mm.

3.2 Laser trace detection

The single corresponding point on the laser trace is detected as the highest intensity pixel in the direction perpendicular to the expected light trace (rows in the image in our case). The position of a pixel with the global maximal intensity is sought. When there is more than one pixel with the same maximal intensity, the position is calculated as the mean of their coordinates. This is the initial estimate of the light trace position with a pixel precision.

However, the sub-pixel precision is needed. The light trace cross-section has the intensity distribution around its maximal value resembling Gaussian,

$$f(x|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} ,$$

where x is the position, μ is mean of the distribution, and σ^2 is the variance. The logarithm of Gaussian distribution is a parabola.

$$\log f(x|\mu,\sigma) = c_1(x-\mu)^2 + \log(c_2) = c_1x^2 + 2c_1x\mu + c_1\mu^2 + \log(c_2) ,$$

where $c_1 = {}^{-1}\!/_{2\sigma^2}$ $c_2 = {}^{1}\!/_{\sqrt{2\pi\sigma^2}}$. The position of the light trace in the sub-pixel accuracy is obtained at the extreme of the parabola.

Saturated pixels on the laser trace cause problems as outliers. They bias the arithmetic mean when fitting the Gaussian/parabola. We omit these points. We fit a parabola to modified data by Least Squares method. The parabola equation writes

$$f(x) = ax^2 + bx + c ,$$

where x is the pixel position, $f(x)$ is the intensity of a pixel at the position x, (a, b, c) are parameters of the parabola. The parabola extreme is located as

$$\frac{\partial f(x)}{\partial x} = 0 \rightarrow x = -\frac{b}{2a} .$$

3.3 Camera-Plane Triangulation

The position of 3D planar points observed by a camera can be reconstructed when the equation of the original plane is known. We use Equation (3.1) extended by the plane equation in a global coordinate system

$$[a, b, c, d] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \vec{a}^\top \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = 0 , \tag{3.3}$$

where $[a, b, c]^\top$ is a plane normal vector and d represents the plane translation from the coordinates origin. After combining Equations (3.3) and (3.1) it holds

$$s \begin{bmatrix} u \\ v \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbb{A} \\ \vec{a}^\top \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} , \tag{3.4}$$

where $\mathbb{A} = \mathbb{K}\,[\mathbb{R}|\vec{t}\,]$. The $[\mathbb{R}|\vec{t}\,]$ is the identity matrix if camera coordinates correspond to global coordinates. We use this method when reconstructing the 3D surface and in the calibration.

3.4 Calibration of the laser projector position

The position of the laser projector for 3D reconstruction must be known. The position of the laser plane in global coordinates is estimated from the laser trace projected on the flat calibration pattern (a chessboard) at different positions, two at least. The calibration pattern from Section 3.1 was used. The transformation $[\mathbb{R}|\vec{t}\,]_{(p,w)}$ from the calibration pattern to the global coordinate system is estimated by solving Perspective-n-Point problem (PnP) [21]. The coordinate system of the calibration pattern assumes that x, y axes are coplanar with the calibration pattern, the z axis is perpendicular to the pattern, and it is directed away from the camera. The calibration pattern plane Equation (3.3) is

$$[0,0,1,0]\;[X,Y,Z,1]^\top = 0\,.$$

The plane is described in chessboard coordinates. The transformation to the global coordinates explores $[\mathbb{R}|\vec{t}\,]_{(p,w)}$

$$\vec{n} = [a,b,c,0]^\top = [\mathbb{R}|\vec{t}\,]_{(p,w)}\,\vec{a}\,,$$

$$\vec{t} = [\mathbb{R}|\vec{t}\,]_{(p,w)}\,[0,0,0,1]^\top\,,$$

$$d = -\vec{n}\cdot\vec{t}\,,$$

where a, b, c and d are coefficients describing the plane, Equation (3.3). This plane is used in Equation (3.4).

Multiple images of the calibration pattern containing the laser trace in different positions are captured. The laser trace is tracked on the pattern surface in image coordinates. The calibration pattern plane equation in global coordinates is estimated. The laser trace points are transformed into global coordinate system using Equation (3.4). All the points of laser trace lay in the laser plane. We can estimate parameters of the laser plane as the approximation of the points if the points are not on a single line. This is satisfied by placing the calibration pattern properly (as shown in Figures 3.3).

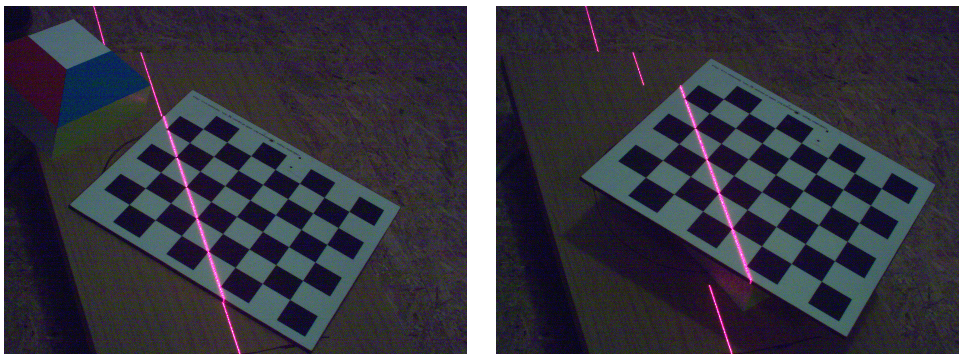
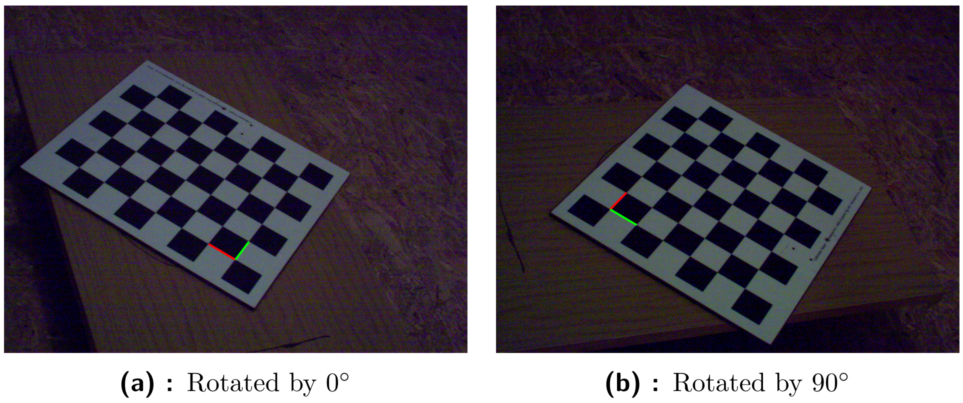

(a) : Lower position **(b) :** Upper position

Figure 3.3: Example of the calibration pattern positioning.

3.5 Calibration of the rotary table

Our last calibration task is to estimate the rotation axis of the rotary table. The calibration pattern is placed on the rotary table, it is rotated, and appropriate images are captured. The example of the calibration pattern in two positions differed by 90° rotations are shown in Figure 3.4. The transformation matrix $[\mathbb{R}|\vec{t}]$ from pattern coordinates to global coordinates using PnP is obtained for each position of the calibration pattern. The rotation axis is estimated from the position of the calibration pattern.



(a) : Rotated by 0° **(b) :** Rotated by 90°

Figure 3.4: Example of the axis calibration patterns. The colored lines show the coordinate system of the calibration pattern in its origin. The x axis is red, and y is green.

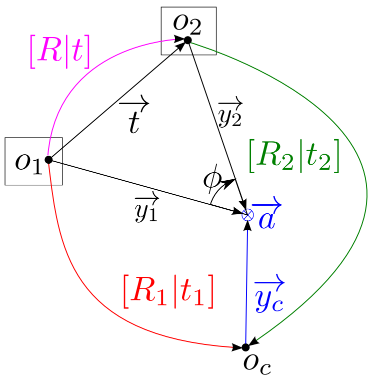


Figure 3.5: Coordinate systems and their relations.

Relations between different coordinate systems are illustrated in Figure 3.5. O_c is the world origin, O_1 and O_2 are origins of the calibration pattern at different rotations, $\vec{a}$ is the direction vector of the rotation axis, $\vec{y_c}$ provides the translation of the rotation axis from the world origin, $[\mathbb{R}_i|\vec{t_i}]$ is the transformation matrix from one coordinate system to another in the direction of an arrow.

The transformation $[\mathbb{R}|\vec{t}\,]$ from O_1 to O_2 is based on the position of the patterns as follows

$$\mathbb{R} = \mathbb{R}_2^{-1}\mathbb{R}_1 \,,$$

$$\vec{t} = \mathbb{R}_2^{-1}(\vec{t}_1 - \vec{t}_2) \,.$$

The rotation axis is described by the direction vector $\vec{a}$ and the translation $\vec{y}_c$. The direction vector $\vec{a}$ satisfies the following equation

$$\vec{a} = \mathbb{R}\vec{a} \,. \tag{3.5}$$

Equation (3.5) can be rewritten as

$$(\mathbb{R} - \mathbb{I})\vec{a} = \vec{0},$$

where $\mathbb{I}$ is a 3×3 identity matrix. We know that $\mathbb{R} - \mathbb{I}$ cannot create null space because $O_1 \neq O_2$. The direction vector $\vec{a}$ must be the eigenvector of the matrix $\mathbb{R}$. We also know that the rotation matrix $\mathbb{R}$ has three eigenvalues, two of which are complex conjugates and the third one is a real number. We seek the real number solution. The eigenvector corresponding to the real solution is named the direction vector $\vec{a}$ in the coordinate system of O_2.

As illustrated in Figure 3.5, the translation $\vec{y}_c$ can be found by moving the vector $\vec{y}_1$ in coordinate system of O_1 by the translation vector $\vec{t}$ and by rotating using the rotation matrix $\mathbb{R}$,

$$\vec{y}_2 = \mathbb{R}\,\vec{y}_1 + \vec{t}\,. \tag{3.6}$$

As O_1 and O_2 are constraint by the rigid transformation, $\vec{y}_1$ in the coordinate system O_1 is equal to $\vec{y}_2$ in coordinate system of O_2. We can adjust Equation (3.6) as

$$\vec{y} = \vec{y}_1 = \vec{y}_2 = \mathbb{R}\,\vec{y} + \vec{t}\,,$$

$$(\mathbb{R} - \mathbb{I})\,\vec{y} = -\vec{t}. \tag{3.7}$$

The Equation (3.7) is underdetermined because these vectors create the circle of possible positions of the rotation origin. The rotation is performed inside the plane with the normal vector $\vec{a}$ and the translation vector $\vec{y}$, which must lay inside of the same plane. These vectors must be perpendicular and satisfy

$$\vec{a}^{\top}\vec{y} = 0 \,. \tag{3.8}$$

We merge equations (3.8) and (3.7):

$$\begin{bmatrix} (\mathbb{R} - \mathbb{I}) \\ \vec{a}^{\top} \end{bmatrix} \vec{y} = \begin{bmatrix} -\vec{t} \\ 0 \end{bmatrix}. \tag{3.9}$$

The Equation (3.9) is now solvable by Moore-Penrose pseudo-inversion [20]. The direction vector $\vec{a}$ and translation vector $\vec{y}$ are in the coordinate system of O_1 or O_2. Both must be transformed to the global coordinates.

3.6 Rotating plane scan about arbitrary axis

The rotation about an arbitrary axis in three dimensions is used when rotating the cut by a specific table rotation angle θ. The formulas come from [22] as a space transformation. This transformation uses the rotation axis obtained in Section 3.5. The transformation $\mathbb{L}$ writes

$$\mathbb{L} = \begin{bmatrix} \mathbb{R} & \vec{t} \\ \vec{0}^{\top} & 1 \end{bmatrix},$$

$$\mathbb{R} = \begin{bmatrix} u^2 + (v^2 + w^2)\cos\theta & uv(1-\cos\theta) - w\sin\theta & uw(1-\cos\theta) + v\sin\theta \\ uv(1-\cos\theta) + w\sin\theta & v^2 + (u^2 + w^2)\cos\theta & vw(1-\cos\theta) - u\sin\theta \\ uw(1-\cos\theta) - v\sin\theta & vw(1-\cos\theta) + u\sin\theta & w^2 + (u^2 + v^2)\cos\theta \end{bmatrix},$$

$$\vec{t} = \begin{bmatrix} (a(v^2 + w^2) - u(bv + cw))((1-\cos\theta) + (bw - cv)\sin\theta) \\ (b(u^2 + w^2) - v(au + cw))((1-\cos\theta) + (cu - aw)\sin\theta) \\ (c(u^2 + v^2) - w(au + bv))((1-\cos\theta) + (av - bu)\sin\theta) \end{bmatrix}.$$

Assuming that θ is an angle of rotation about a line though (a, b, c) with a unit directional vector (u, v, w). The final transformation is

$$\vec{x}' = \mathbb{L}\,\vec{x},$$

where $\vec{x}$ is a point before and $\vec{x}'$ is the point after rotation about the rotation axis. Both of which are described in homogeneous coordinates.

Chapter 4
Implementation

We decided to build experimental LPRF with the motorized, and the computer controlled rotary table. Our simple construction provides support for the camera and laser plane illumination source, i.e. the laser diode with the cylindrical lens.

4.1 LPRF Hardware

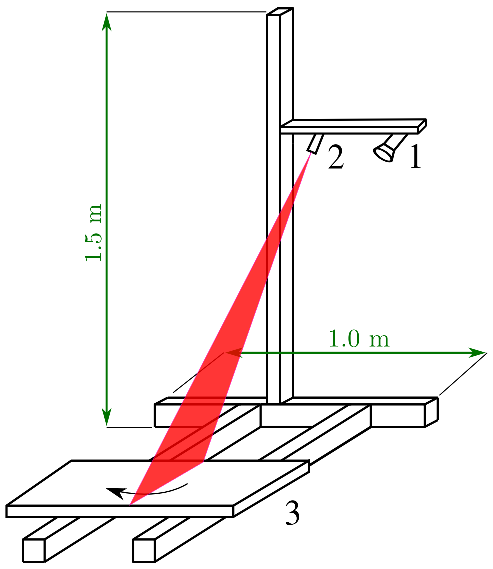


Figure 4.1: Proposed construction. (1) The camera, (2) the laser plane illuminator, and (3) the rotary table.

The sketch of our construction is in Figure 4.1. The construction contains three major parts, (1) the camera, (2) the laser plane illuminator, and (3) the rotary table. All parts are connected by a supportive construction from the aluminium alloy [17]. The following construction was chosen because of its stability and suitability in terms of modularity.

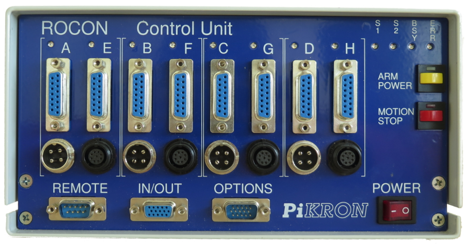
We used a BASLER daA2500-14uc camera [2] (Figure 4.2a) with the resolution 2592 × 1944 px and with interchangeable lenses by TAMRON.

We chose 12 mm F/1.4 lens [14] (Figure 4.2b), for our experiments. The laser plane is generated by a programmable semiconductor laser the with the cylindrical lens made by COHERENT, type StringRay [6] (Figure 4.2c).

(a) : BASLER daA2500-14uc camera Courtesy [2].

(b) : 12 mm F/1.4 Tamron lens. Courtesy [14].

(c) : COHERENT stingray. Courtesy [6].



(d) : MARS 8, control unit. Own photo.

Figure 4.2: LPRF hardware.

The rotary table is driven by DC motor with gearing. An optical incremental rotary encoder with 2,500 periods per revolution is connected directly to the main rotation axis of the table. The position is set by a feedback controller. The rotary table is controlled by the MARS 8 unit from PiKRON s.r.o. [16] (Figure 4.2d). The controller is able to set the rotation angle of the table with the precision of a quarter of the period (10,000 positions per one revolution).

The scanning area has a circular base with diameter approximately 30 cm and height 15 cm. The scanning area size depends on the used camera lens.

4.2 LPRF Software

The LPRF is implemented as the Python package. The package is modularized to different sub-packages for camera control, rotary table control, laser tracking, calibration, visualization and precision assessment.

The camera control package (driver) is able to set the camera capturing parameters and collect images. The used BASLER camera is provided with C++ library. Consequently, the camera control package is just a Python wrapper for BASLER C++ library. The package converts BASLER camera into the OpenCV2 camera. If a different camera is used, which supports OpenCV2, this package can be omitted.

The rotary table control package enables controlling the rotary table. The package allows setting up, rotating and reading data from the rotary table. Our implementation uses rotary table connected via MARS 8 control system. Our rotary table uses USB communication allowing to perform set of predefined commands.

The laser tracking package tracks the laser trace in the image. The method was described in Section 3.2. The package uses lens distortion coefficients, which were explained in Section 3.1. This allows correcting the position of the laser trace.

The calibration package calibrates our LPRF. The camera calibration was described in Section 3.1. The package also finds the laser trace position, Section 3.4 and the rotation axis, Section 3.5.

The visualization package uses the OpenGL library for plotting the scan point cloud. We chose this visualization technique because of the huge number of measured 3D points.

The Python code for precision assessment is in its α version only. It was designed and written by a student Mr. Uran Okudomi[1] during his two and half month stay with us in fall 2016. The algorithm was sped up by me, but it still uses his visualization.

[1]Visiting master student from Tokyo University of Agriculture and Technology.

Chapter 5
Experiments

Having our experimental LPRF, we captured several scans of different objects. The first object is the object ‘cube’, which is a frustum with the cubic base, Figure 5.1. The cube was used later for evaluating the precision of 3D reconstruction. The second test object is a piece of fabric with a freeform surface, a dish towel, Figure 5.2. Furthermore, we created the scan of the calibration pattern, Figure 5.3a, Abraham Lincoln’s bust, and the folded paper.

We designed and performed the experiment evaluating the measurement precision by scanning a flat surface. We chose the surface of a rotary table, which is flat and perpendicular to the table rotation axis. The other issue is the possibility to increase precision. The detected calibration points do not cover the whole calibration pattern. OpenCV2 calibration does not take into account the first and last row of chessboard squares, which makes the calibrated region smaller. If we constrain the measurement to the calibrated region, the precision is increased. We will demonstrate it practically.

The last experiment, we designed and performed in the cooperation with Vladimír Petrík, was the scanning of cloth strips, the result will be later used by Vladimír Petrík for guessing the kinematic parameters of strips. This experiment follows our motivation towards scanning freeform objects.

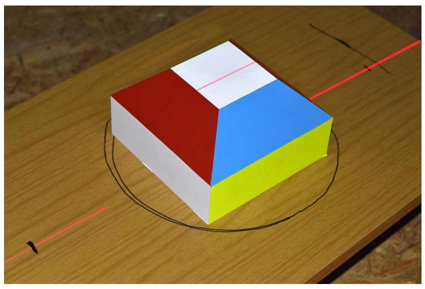
5.1 Measured object

We saw the ‘cube’ in Figure 5.1a. We know its dimensions, the base square is 15 cm $\times$ 15 cm and height is 9 cm. We visualize its scans in Figure 5.1b. ‘Cube’ sides have differently colored surfaces. The laser trace irradiated in the camera direction differs for varied colors. Notice this phenomenon on 3D points corresponding to the blue surface plane, where a significant collection of points is missed (the white region). This contrasts with the red ‘cube’ surface, which is covered by 3D points almost entirely.

Figure 5.2a shows the object with a freeform surface, a wrinkled dish towel. Its scan is in Figure 5.2b. The laser trace does not reach all the folds. The camera cannot see under all folds either. This is caused by the dish towel self-occlusion. Consequently, the void (white, empty) spots appear in the scan.

Figures 5.3 show the calibration pattern in Figure 5.3a and its scan in Figure 5.3b. We can demonstrate, how the color of the surface effects the laser plane reflection to the camera. The light on the white surface is well reflected to the camera and correctly captured by the sensor. On the other hand, the light on the black surface ends up as a thrashed sample and creates void spots visible on the scan in Figure 5.3b.

Figures 5.4 show the scan of Abraham Lincoln's bust in Figure 5.4a, the scan of the folded paper in Figure 5.4b.


(a) : Cube, the object with known diameters to assess the measurement precision.

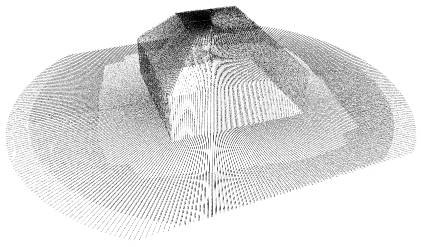
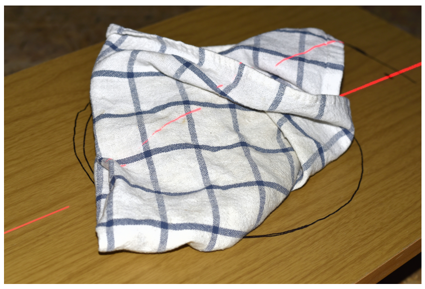

(b) : The cube scan.

Figure 5.1: The cube


(a) : The object with a freeform surface, the dish towel.

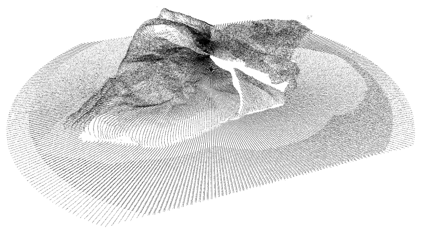
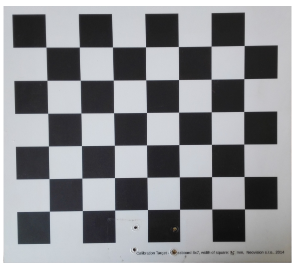

(b) : The dish towel scan.

Figure 5.2: The dish towel


(a) : The calibration pattern picture.

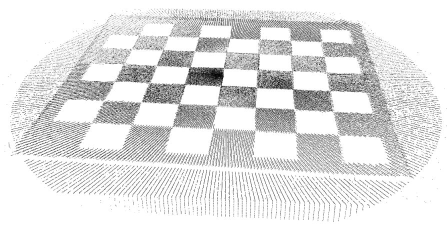
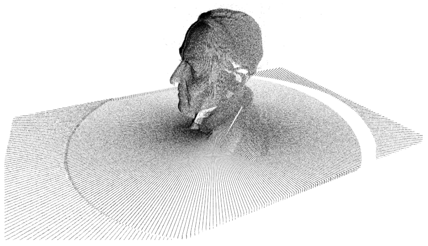
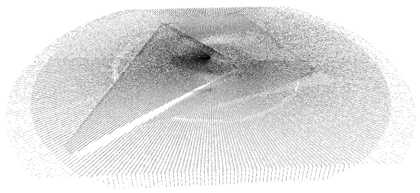

(b) : The scan of calibration pattern.

Figure 5.3: The calibration pattern.



(a) : The scan of 3D printed Abraham Lincoln's bust.



(b) : The scan of folded paper.

Figure 5.4: Scans of the stuff.

5.2 Evaluating the precision of a plane scan

The calibrated region is shown in Figure 5.5. Magenta dots symbolize the corners of the calibration pattern, a light blue line is a laser trace, and horizontal dark blue lines separate image into the top part, middle calibrated region, and the bottom part.

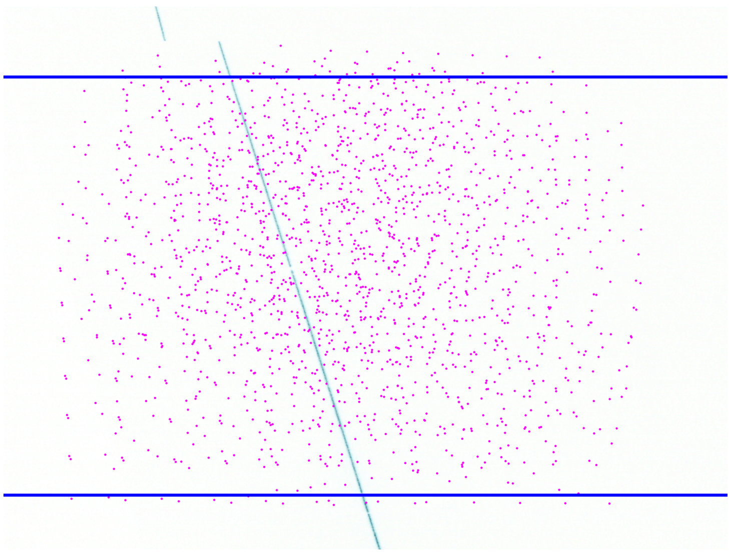


Figure 5.5: Calibration points in magenta. The calibrated region is between dark blue lines. The laser trace is in light blue.

OpenCV2 calibration requires seeing the entire calibration chessboard pattern with one extra row and column of pattern segments around. In our case, we have been unable to reach the bottom and top part of the image with this type of calibration. The bottom and the top part is not covered with measured points. Moreover, the rotation axis of the table is in the middle of the image. Therefore, we receive two points for the similar spot in 3D when we rotate the table all the way around.

The point cloud with a large amount of points is received from LPRF. The plane is fitted into the point cloud using RANSAC [26], and support points[1] are collected. The plane equation is obtained using the least square method on support points. We define contiguous patches covering the whole plane. Support points, which lay inside one patch, define a patch point set. The distances between each point in one patch point set and plane are calculated. We are able to calculate the standard deviation (std) for each patch point set from distances and symbolize the precision as std.

We can see the precision evaluation of a rotary table planar surface without cropping to the calibrated region of the scan image in Figure 5.6 and with cropping in Figure 5.7. The precision is symbolized as a standard deviation from the plane.

We can observe the circle with hight std in the middle of Figure 5.6, which is caused by the uncalibrated region in the bottom of the figure. The precision is given in mm, see the color scale on the right side of the picture. The standard deviation of the whole plane was 0.49 mm. Furthermore, the top is not also calibrated as we can see in Figure 5.5. Hence, we cropped the bottom and the top of the image. We can see the crop bottom and top scan picture in Figure 5.7. Even though, the scan area is smaller it is also more precise than before. The standard deviation of the whole plane was 0.28 mm.

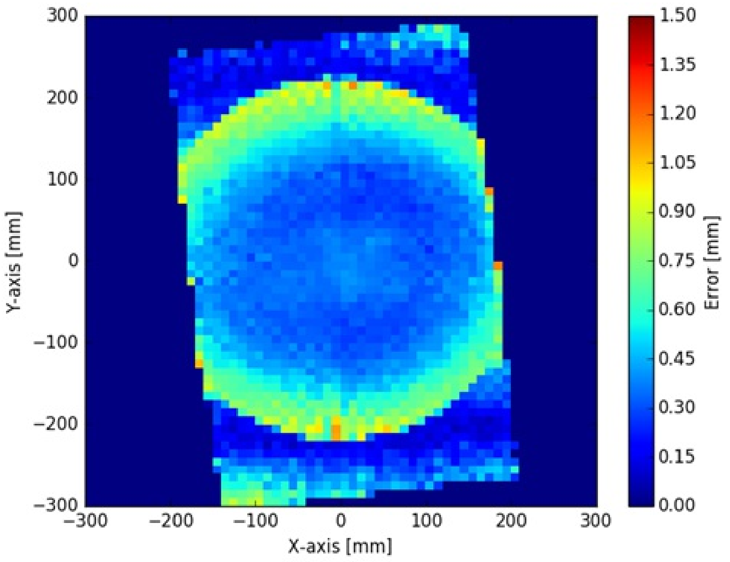


Figure 5.6: Plane, no cropping to calibrated region.

[1]Points which voted for the plane selection during RANSAC algorithm.

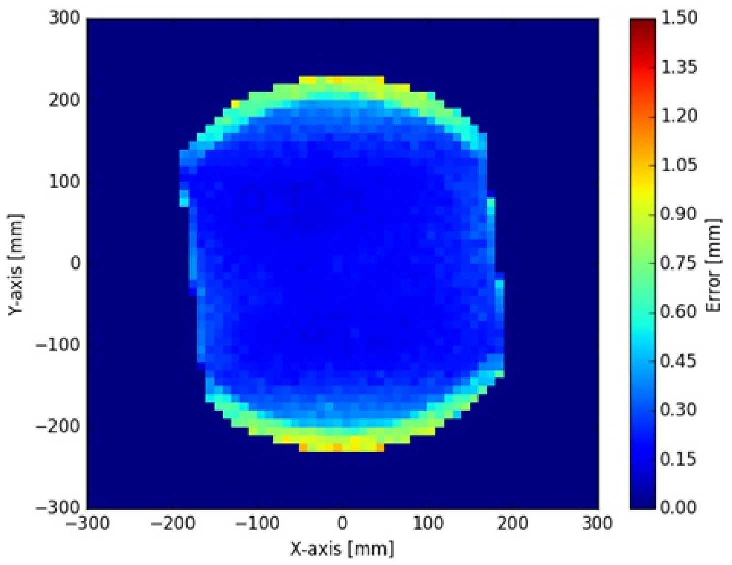


Figure 5.7: Plane, cropped to calibrated region.

5.3 Precision of the cube scan

The cube scan in Figure 5.1 was used to the evaluation of the range-finder accuracy. The dimensions of the cube are known. We used multi-run RANSAC for fitting 10 planes to the scan. One plane represents the surface of the rotary table. Other nine planes create the surface of the cube. The precision evaluation was done semi-automatically.

The cube sketch is in Figure 5.8. The cube height is 90 mm, The cube can be separate into two parts, the bottom, and the top. The bottom part is a cuboid with a square base with an edge size of 150 mm, and with height of 50 mm. The top part is the frustum with a square base with 150 mm edge length. The top plane is a square with 70 mm edges. Angles between the bottom square and side planes are 45°.

Vertex tuples (ordered sets) $\mathbb{V}_b = (V_{b1},\ V_{b2},\ V_{b3},\ V_{b4})$ create a bottom part of the cube, $\mathbb{V}_m = (V_{m1},\ V_{m2},\ V_{m3},\ V_{m4})$ defines a middle part, and $\mathbb{V}_t = (V_{t1},\ V_{t2},\ V_{t3},\ V_{t4})$ are a top part of the cube. Each two following vertexes in the vertex tuple define the edge of the cube. Tuples are connected via edges between vertexes with the same number. Each two edges connected with one point defines a plane. Planes defined by vertexes from each vertex tuple are parallel.

We start with the evaluation of angles between each plane. Each angle between two planes was calculated as

$$\theta = \arccos(\vec{n}_1 \cdot \vec{n}_2),$$

where θ is the angle between planes, and $\vec{n}_i$ is a unit normal vector of each plane.

We start with the bottom part of the cube, the cuboid. All planes with the common edge in the cuboid, and the plane defined by vertex tuple $\mathbb{V}_t$ and sides

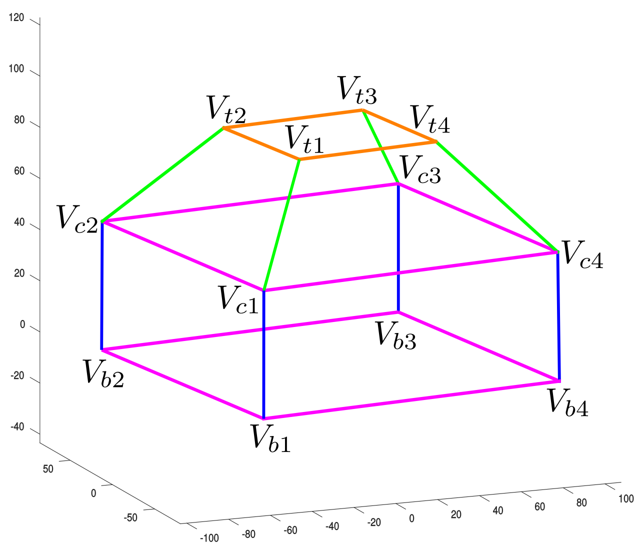


Figure 5.8: The cube sketch with named vertexes. Edge colors coorespond to distances between point. Orange ones are 70 mm, magenta ones are 150 mm, blue ones are 50 mm, and green ones ≈ 69.282 mm.

of the cuboid are perpendicular (90°). All of the opposite sides are parallel, and theirs direction vectors have the opposite direction (180°). The top part of the cube, the frustum, has angles between base or top and sides 45°, esp. 135°. Opposite planes on the side of the frustum are perpendicular, and angles between connected sides are 60°.

After calculation of the angle θ between each two planes, all angles were close to proposed ones. We grouped the angles by their proposed value and calculate its mean and standard deviation (std) of each group. Table 5.1 shows results.

proposed angle [°]	**mean of** θ [°]	**std of** θ [°]
45	44.8731	0.1682
60	59.9977	0.2860
90	89.9339	0.2426
135	134.8721	0.2054
180	179.5857	0.2014

Table 5.1: Grouped angles between planes.

Following the evaluation of angles, we evaluate the length of edges of the cube. The length of edges are shown as colors in Figure 5.8. Orange ones are 70 mm, magenta ones are 150 mm, blue ones are 50 mm, and green ones ≈ 69.282 mm.

The length of the edge is calculated as Cartesian distance between two vertexes which defines this edge. Each vertex is defined as an intersection of 3 planes (Courtesy [27]).

Vertexes in vertex tuples $\mathbb{V}_b$, and $\mathbb{V}_t$ are defined by using only 3 planes and have one solution. Vertexes in $\mathbb{V}_c$ are determined using 4 planes (2 cuboid and 2 frustum sides), and have 4 solutions for each vertex in the tuple. The mean position of those 4 vertexes is used to find only one solution for each vertex in vertex tuple $\mathbb{V}_c$. Having all sets, we can find distances between vertexes using the Cartesian distance:

$$dist = \sqrt{(V_{x1} - V_{x2})^2 + (V_{y1} - V_{y2})^2 + (V_{z1} - V_{z2})^2},$$

where $V_i = (V_{xi}, V_{yi}, V_{zi})$ are vertex coordinates. We calculate an

$$error = dist - dist_{exp}\,,$$

where $dist_{exp}$ is expected the length from Figure 5.8. Calculated distances and errors are shown in Table 5.2. The results do not deviate from expected values for more than 1 mm.

V_1	V_2	$dist_{exp}$ **[mm]**	$dist$ **[mm]**	$error$ **[mm]**
V_{b1}	V_{b2}	150.00	149.68	-0.32
V_{b2}	V_{b3}	150.00	150.05	0.05
V_{b3}	V_{b4}	150.00	149.72	-0.28
V_{b4}	V_{b1}	150.00	149.50	-0.50
V_{c1}	V_{c2}	150.00	149.22	-0.78
V_{c2}	V_{c3}	150.00	149.82	-0.18
V_{c3}	V_{c4}	150.00	149.19	-0.80
V_{c4}	V_{c1}	150.00	148.94	-1.06
V_{t1}	V_{t2}	70.00	69.27	-0.73
V_{t2}	V_{t3}	70.00	70.27	0.27
V_{t3}	V_{t4}	70.00	69.20	-0.8
V_{t4}	V_{t1}	70.00	69.08	-0.92
V_{b1}	V_{c1}	50.00	49.91	-0.09
V_{b2}	V_{c2}	50.00	50.30	0.30
V_{b3}	V_{c3}	50.00	50.55	0.55
V_{b4}	V_{c4}	50.00	50.21	0.21
V_{t1}	V_{c1}	69.28	69.53	0.25
V_{t2}	V_{c2}	69.28	68.69	-0.59
V_{t3}	V_{c3}	69.28	68.87	-0.40
V_{t4}	V_{c4}	69.28	69.46	0.17

Table 5.2: The length of edges.

All of the expected values come from the cube drawing, and its exact dimensions are unknown. The evaluation was done semi-automatically, as we said before. All planes were fitted using multi-run RANSAC, generating support points for each plane. The plain equation is obtained by using the least squares method on support points. The assignment of the planes intersections and cube vertexes must be done manually. Dimensions of the cube and angles between cube planes are evaluated manually too.

5.4 Cloth strips manipulation

This experiment follows our motivation and uses the CloPeMa robot. The range finder was used to measure the position and the shape of cloth strips, which is folded by the robot. The configuration is in Figure 5.9. The data from the measurement will be used to guess parameters of the kinematic description of the strip by Vladimír Petrík in his research.

Each strip lays on the table and is held by the robot as is shown in Figure 5.9. The robot performs an experiment in different height over the table. The robot moves the strip in the direction of a green arrow, away from LPRF, holds it, and forms a fold. It returns to the position in Figure 5.9 afterward, tightens the strip with its weight, and performs the experiment again in the different height. In the meantime, LPRF collects images for the measurement of a shape of the strip.

The experiment was performed in seven different heights, from 35 cm down to 5 cm, and with five different cloth strips. Each experiment (seven different heights) took about 330 seconds with a scan rate roughly about 6 frames per second. Frames were processed after the experiment. The processed data during single fold and for 30 cm, and 15 cm height are shown in Figures 5.10, and 5.11. Both figure sets show the process of a fold creation by the robotic arm.

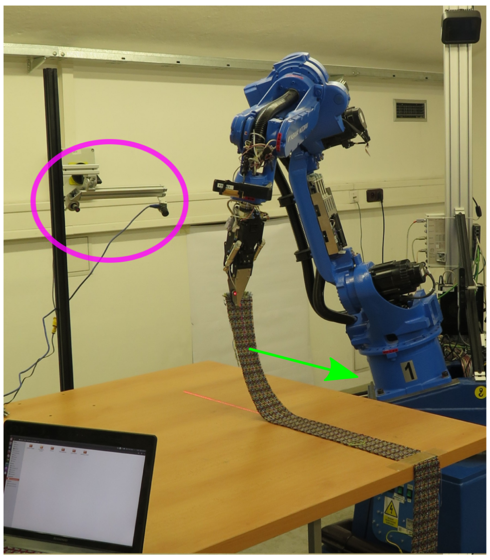
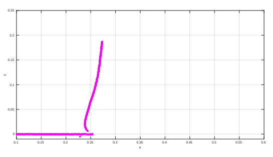
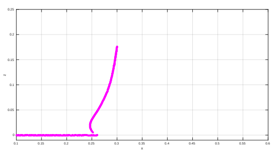


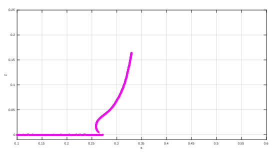
Figure 5.9: LPRF (magenta circle) setup with CloPeMa robot. Green arrow shows the direction of movement for the fold creation.



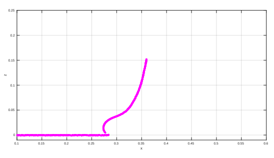
(a) : Time 51s.



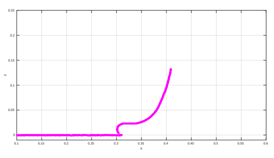
(b) : Time 52s.



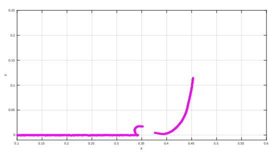
(c) : Time 53s.



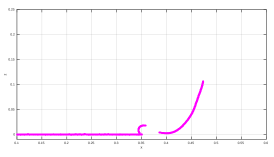
(d) : Time 54s.



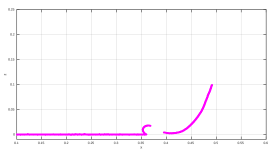
(e) : Time 55s.



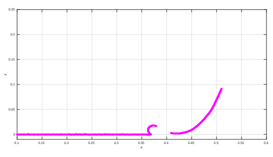
(f) : Time 56s.



(g) : Time 57s.

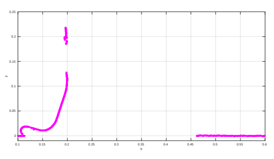


(h) : Time 58s.

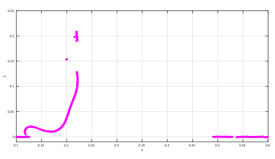


(i) : Time 59s.

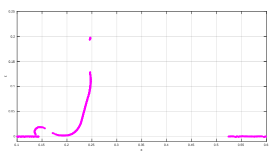
Figure 5.10: The scan of the strip, cloth no. 1. Height 35 cm.



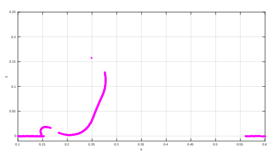
(a) : Time 233s.



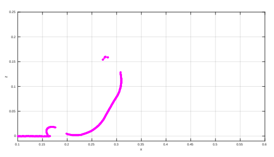
(b) : Time 234s.



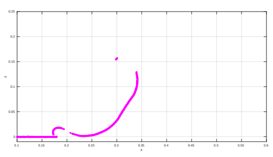
(c) : Time 235s.



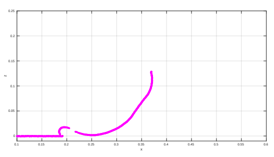
(d) : Time 236s.



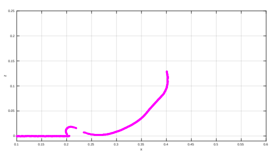
(e) : Time 237s.



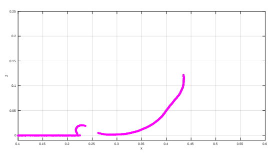
(f) : Time 238s.



(g) : Time 239s.



(h) : Time 240s.



(i) : Time 241s.

Figure 5.11: The scan of the strip, cloth no. 1. Height 15 cm.

Chapter 6

Conclusions and future work

The task was to create a duplicable and reconfigurable laboratory LPRF hardware and its software. This thesis summarizes all the components, software design, calibration process and the evaluation of precision.

We built LPRF and evaluated it experimentally. We provide the methodology and implemented tools for precision assessment. The evaluation is in the first version so far. Part of the evaluation must be done manually.

We set four tasks in Section 1.2. First two were the developing and implementation of LPRF. The third one was the method testing, and the last one was to document processes and put it into the public domain.

Developed methods and implementation. The LPFR components come from our laboratory. The components are a camera, a laser plane emitter, and a rotary table, see Section 4.1. The table is not required if a single cut is needed, see Section 5.4. The scanned area is based on used camera and lenses. Different camera lenses can be used to increase, or decrease it. We use the cylindrical scan area with radius and height of 150 mm in this thesis.

The software provides a calibration, a basic visualization, and a simple plane estimation. The camera calibration process mainly uses OpenCV2 calibration with a chessboard pattern. We introduce a method for finding the rotary table axis in the camera using the transformation matrices. We provide the mathematics needed for the triangulation between the camera and a laser ray. Moreover, we create a simple visualization using OpenGL, and a simple plane estimation using RANSAC [26] for the evaluation performed by the visiting student Mr. Uran Okudomi. This paragraph satisfies our first two tasks.

Testing of developed methods. We performed a bunch of experiments using LPRF. First experiments were scans of objects, and stuff from the lab, see Section 5.1. The second and the third experiment, see Section 5.2, and Section 5.3, gave a basic idea about the precision of LPRF. The standard deviation of points of a single measured plane is 0.28 mm. The reference object, a cube, has the worst standard deviation of angles 0.286°, and the largest deviation of the edge length is 1.06 mm. The evaluation stems from the scanned area. Finally, we used LPRF as the data acquisition for the guessing parameters of the kinematic description of the cloth strip. The precision of our LPRF corresponds to the used construction and physical capabilities of it. The task of testing the developed method is satisfied by this paragraph.

Public availability. The last task was to put everything into the public domain. We created the software tool and came up with the construction described in paragraphs above.

The software tool is capable of estimating camera parameters, calibrating the relations between the camera, and the laser plane emitter, and finding the rotation axis of the rotary table. These parts work and can be used. However, the code is in the state of testing, and some instabilities can occur. The tool contains the basic OpenGL visualization and RANSAC plane estimation for the evaluation. These parts of the code are in α version only can be unstable. The evaluation process is done semi-automatically and is not publicly available yet. The assignment between cube vertexes and points of plane intersections is done manually. Dimensions of the cube and angles between cube planes are evaluated manually too.

We put our code into the public domain on GitLab[1]. GitLab contains a basic documentation which must be adjusted.

Future work. The ideas for future work are:

- **Short term:** Extend the calibration to planar movement; It is easier than rotary movement; We need to build a testing device in the lab first.
- **Long term:** (1) Support camera calibration, which can cover the whole image calibration chessboard image; OpenCV3 has a calibration pattern/tool allowing it. (2) Vectorized tracking of the light trace in the direction of the trace, which should increase precision.

[1]GitLab: https://gitlab.ciirc.cvut.cz/cmirajak/laser_plane_scanner

Bibliography

[1] AQSense,SL., 3d machine vision library - imaginglab, checked May 16, 2017. http://www.aqsense.com/products/3d-industrial-machine-vision-library.html.

[2] BASLER AG, USB camera daa2500-14uc. Datasheet, checked May 16, 2017. http://www.baslerweb.com/en/products/cameras/area-scan-cameras/dart/daa2500-14uc.

[3] bq, Ciclop, checked May 16, 2017. http://diwo.bq.com/en/presentation-ciclop-horus/.

[4] Cognex Corporation, 3d displacement sensors, checked May 16, 2017. http://www.cognex.com/products/machine-vision/ds-1000-displacement-sensor-laser-profiler/.

[5] Microsoft Corporation, Kinect 2 motion sensing input devices by Microsoft, checked May 16, 2017. http://www.xbox.com/cs-CZ/xbox-one/accessories/kinect.

[6] COHERENT Inc., Stingray, checked May 16, 2017. https://www.coherent.com/lasers/laser/machine-vision-structured-light-lasers/stingray-lasers.

[7] FARO Technologies UK Ltd, Design scanarm, checked May 16, 2017. http://www.faro.com/products/3d-documentation/faro-design-scanarm/overview.

[8] FreeLSS. 3d scanning package, checked May 16, 2017. https://github.com/hairu/freelss.

[9] 3D SYSTEMS Corporation, GEOMAGIC, 3d scanning software, checked May 16, 2017. http://www.geomagic.com/en/.

[10] HORUS. 3d scanning package, checked May 16, 2017. http://horus.readthedocs.io/en/release-0.2/index.html.

[11] Hewlett-Packard Company, 3d scanner, checked May 16, 2017. http://www8.hp.com/us/en/campaign/3Dscanner/overview.html.

[12] M. LLC. Atlas 3d, checked May 16, 2017. https://www.kickstarter.com/projects/1545315380/atlas-3d-the-3d-scanner-you-print-and-build-yourse.

[13] MVTec Software GmbH. Halcon 3d vision, checked May 16, 2017. `http://www.mvtec.com/services-solutions/technologies/3d-vision/`.

[14] TAMRON US, Inc., 2/3 12mm f/1.8 with lock, checked May 16, 2017. `http://www.tamron-usa.com/IO/IndOp/fixed_focal.php`.

[15] OpenCV. Docs, checked May 16, 2017. `http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html`.

[16] PiKRON s.r.o., Mars 8 unit, checked May 16, 2017. `http://www.pikron.com/pages/products/motion_control/mars_8.html`.

[17] item Industrietechnik und Maschinenbau GmbH, item parts, checked May 16, 2017. `http://www.item24.de`.

[18] Šonka, M. and Hlaváč, V., and Boyle, R.. Image Processing, Analysis and Machine Vision. Cengage Learning, 4rd edition, 2015, ISBN: 978-1-133-59360-7.

[19] P. F. Sturm and S. J. Maybank. On plane-based camera calibration: A general algorithm, singularities, applications. In *CVPR*, pages 1432–1437. IEEE Computer Society, 1999, Print ISBN: 0-7695-0149-4.

[20] E. Boman. The moore-penrose pseudoinverse of an arbitrary, square, k -circulant matrix. *Linear and Multilinear Algebra*, 50(2):175–179, 2002, Print ISSN: 1083-4419.

[21] S. Li, C. Xu, and M. Xie. A robust o(n) solution to the perspective-n-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1444–1450, July 2012, Print ISSN: 0162-8828.

[22] G. Murray. Rotation about an arbitrary axis in 3 dimensions, checked May 16, 2017. `https://sites.google.com/site/glennmurray/Home/rotation-matrices-and-formulas`, June 2013.

[23] D. Pagliari and L. Pinto. Calibration of kinect for xbox one and comparison between the two generations of microsoft sensors. *Sensors*, 15(11):27569–27589, 2015, Print ISSN: 1424-8220.

[24] T. Pajdla. Laser plane range finder, the implementation at the cvl, 1995, checked May 16, 2017. `http://cmp.felk.cvut.cz/ftp/articles/pajdla/lprf.pdf`

[25] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *ICCV*, pages 666–673, 1999, Print ISBN: 0-7695-0164-8.

[26] Fischler, Martin A. and Bolles, Robert C.. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography In *Communications of the ACM*, 1981, ACM: 24:381-395.

[27] Weisstein, Eric W. "Plane-Plane Intersection." From MathWorld–A Wolfram Web Resource, checked May 16, 2017. `http://mathworld.wolfram.com/Plane-PlaneIntersection.html`