VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Diploma Thesis Typesetting Demo

Ukázka sazby kvalifikační práce

Bc. Pavel Mandrla

Semestral project

Supervisor: Mgr. Ing. Michal Krumnikl, Ph.D.

Ostrava, 2021

Contents

Chapter 1

Introduction

The 3D printing industry has experienced a giant boom in the last fifteen years. Intensive research and inovation in this area have greatly expanded its posibilities, and 3D printing is therefore more and more often used by big industrial companies to manufacture high-tech products.

However the roots of this industry come from open source projects, driven by a passionate community of hobbyists and tinkerers, and that is greatly reflected in todays 3D printer market. Many companies do not focus their products on large industrial subjects, but on public. That means that the price of 3D printers has gone down, while their accessibility and capability highly increased, and it is not uncomon for a household to own a 3D printer.

With these devices becoming more widespread comes a great demand for high quality 3D models. These can be created either by using CAD software, which creates hig precision 3D model, but has a very steep learning curve, or by creating a 3D scan of an already existing object.

As the price of an industrial 3D scanner can be very high, the 3D printing community had to turn to more cost friendly alternatives. Devices, like the Kinect from Microsoft can be used, when scanning larger object, but when it comes to smaller ones, the quality of the resulting 3D model is not sufficient. Several open source alternatives exist for scanning small objects. They utilize easily sourced off the shelf parts and are therefore much more accesible to public. Whilst these solutions are not utilising as complex technologies, as their industrial counterparts, the quality of the resulting 3D prints can still be highly satisfactory.

Chapter 2

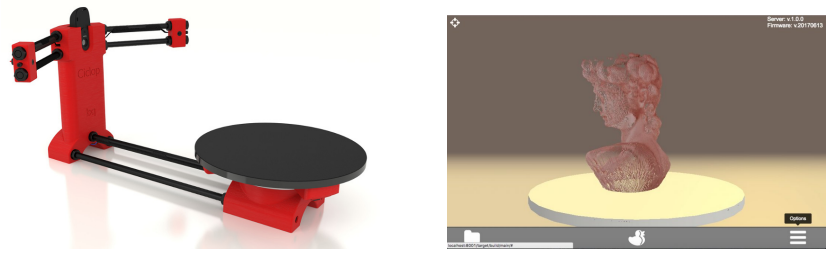
Open source solutions

3D scanners can be devided into two categories, based on the technology, that is used scan the object and create the 3D mesh. These catgories are contact based, which require physical contact with the scanned object, to measure the location of each point on the obejcts surface, and non-contact based, which estimate the position of each point but by using passive scanners, such as cameras. [1] The latter are much easier to design and construct and therefore are much more prevelant within the open source community. They can be further divided into two subcategories, which are passive and active.

2.1 Active scanners

Active sensors actively emit radiation such as light towards the surface of the object to scan it. It can be used similarly to contact based scanners, but instead of having to use physical probe to measure the position of the surface, the distance can be measured by a time-of-flight sensor, which measures the time it takes the emmited light to fly to the object and bounce back.

A different approach is the use of structured light. In this approach, light patterns are shined upon the surface, which is then captured by the camera. These pattern enhance the contours of the object and help with the triangulation of the surfaces points.



(a) (b)

Figure 2.1: (a) BQ Ciclop (taken from [2]) (b) 3D scan creted using FabScan (taken from [3])

Examples of popular projects, that use active scanning to create the 3D mesh, are FabScan, BQ Ciclop and FreeLSS. [2, 3, 4] All of these projects are based on the same technique, where the scanned item is placed on a turntable, which rotates it along the Z axis, which allows a camera to capture the object from different angles. Meanwhile a planar laser is shining towards the turntable, intersecting the subject. The line created by the laser on the objects surface highlights its outline. Because the outline illuminated by the laser is brighter, than the ambient ligt, which illuminates the rest of the object, it can be easily extracted from the image. Based on the position of the turntable, the position of each point along the outline can be triangulated. The points from all the scanned pictures are then stiched together to create the final mesh.

2.2 Passive scanners

Passive scanners use visual information acquired by cameras to create the resulting 3D mesh. The technique used by passive scanners is called photogrammetry or Structure from Motion (SfM). It calculates positions of the objects points in space based on photographs of the object taken from multiple angles. [5] Photogrammetric reconstruction software compares individual images and tries to find tie points, which represent the same point in space between two images. Based on these tie points, the camera position, from which each photo was taken, is calculated and after that the position of each point of the object can be determined.

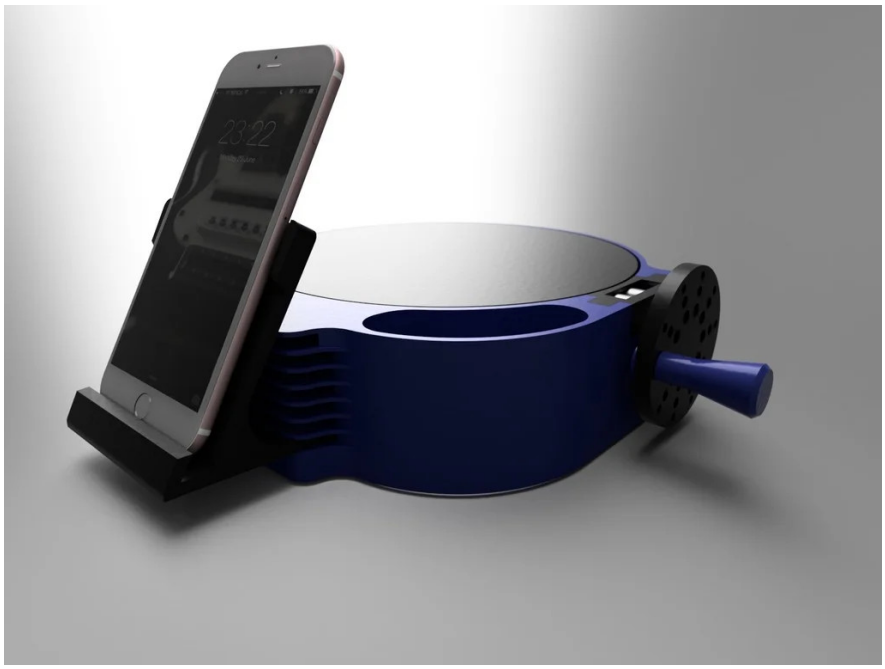


Figure 2.2: The $30 3D scanner (Taken from [6])

Because even cameras available in todays mobile phones are sufficient for high quality photogrammetric 3D reconstruction, this technique is very accessible and cost efficient. Since capturing the images by hand can be a quite tedious task, as the reconstruction requires many images taken from various angles, the available scanners focus on automation of this process.

2.2.1 OpenScan

OpenScan is a project created by Thomas Megel in 2018. It is a device, that automaticly takes photos of the scanned object from multiple angles, by rotating it along two axis. These photos can be fed into a photogrammetry reconstruction software, to create the resulting 3D mesh.

It supports many different types of cameras, so it is very inviting to all types of users. As cheaper option, it allows for the use of phones or webcams, to scan the object, but if the user desires higher quality scans, the scanner is also compatible with DSLR cammeras. It is powered either by a Raspberry Pi or an Arduino, and multiple variations of this scanner exist. The develoment of this project is still very active and it receives regular updates.

Figure 2.3: OpenScan (Taken from [7])

Chapter 3

My contribution //TODO -> rename

I have chosen to work with the OpenScan project because of multiple reasons. First I was wery interested in photogrammetry and its use for 3D reconstruction. It alseo seemed to me, that the project has a lot of potential to become succesfull even within the more casual part of the 3D printing comunity, as it is easy to setup and has a simple user interface. It alse uses the more simplistic aproach of using photogrammetry, which requires less calibration than the use of structured light, so it is very user friendly. I also liked that the scanner does not just rotate the object along the Z axis, but allows the camera to capture the object from the top and bottom.

Where I saw room for improvement, was that the output of the scanner is not a finished 3D representation of the scanned object, such as an .stl file, but just a set of images that the user has to process himself with an additional tool, such as Meshroom.

The author of the project has since started working on his own solution to this problem and in February launched a beta of "OpenScan Cloud", which is a cloud service, that automaticly processes the scanned images and produces the resulting 3D mesh. [8]

I have decided to use a slightly different approach, and instead of sending the scanned pictures onto a server, where the processing and asembly would be done, I would do the required calculation locally. The only problem with this approach, is that the Raspberry Pi, which is used by the OpenScan project, is not powerful enough, to do the required processing. Because of this reason, I've decided to replace the Raspberry Pi with a more powerful alternative, which is the Jetson Xavier NX by Nvidia. Its more powerful processor, larger memory and CUDA capable graphics card allowed me to process the scanned data locally, without the need to send it to a server for 3D mesh reconstruction.

3.1 Evaluation of the existing solution

With the change of the platform, some problems with the current solution became apparent. Even though the software for the scanner was created using platform independent technologies, the code

itself was writen without the prospect of running on a different platform. The main cause of this was in my opinion the use of the programming tool Node-RED. It is a graphical programming languague built on Node.js, which allows the user, to connect together different nodes that represent parts of the code. [9] Some nodes can for example contain JavaScript code, that will be executed, when the node gets activated.

The author of the OpenScan project decided to use an extension, which allowed him to use Python instead of JavaScript. This solution did not allow for easy reuse of the python code, and so the code for each functionality had to be rewritten inside each node, that was using that functionality.

I also belive, this was the reason, why the author decied to store each parameter of the scanners configuration in a separate file. Whenever the software needs to know the value of a parameter, such as the number of photos, the scaner should take, or the number of the GPIO pin used by a stepper motor, it loads a file, which contains the parameters value. Because the paths to these configuration files are hardcoded in the codebase, any changes to the configuration code is very difficult.

For these reasons, the code structure of the scanners software became quite bloated and chaotic, which must make maintanace of this project a hard task. It also means, that moving the codebase to a different platform would be quite complicated.

3.2 Used technologies

Because of the reasons stated in the previous section, I have decided to create my own software, which would controll the scanner. As my languague of choice, I used Python 3, because the Jetson provides an easy to use interface to use the GPIO pins, with Python. It also allowed me to reuse some parts of the original OpenScan software, such as the code that controlls the stepper motors, which I adjusted to suite my needs. I also replaced the many configuration files, with a single JSON document, which contains the values of each parameter and is loaded into memory after the start of the scanning procedure.

My implementation was mainly concerned with the scanning process itself, as I decided not to reimplement some features of the openscan project, such as the web-based user interface, or the use of a SAMBA protocol to easily share the scanned files, as they were not crutial for my project.

//TODO - > elaborate on my implementation

The move from Node-RED to Python3 made the codebase much more compact and easily maintainable. I belive, that this would be the right move for the project in the future, because it would make addition of new feateures and maintanance much easier.

Sources

1. ENGELMANN, Francis. FabScan Affordable3D Laser Scanning of Physical Objects. [N.d.]. Available also from: https://hci.rwth-aachen.de/publications/engelmann2011a.pdf.
2. *Ciclop*. 2017-07. Available also from: https://reprap.org/wiki/Ciclop.
3. *FabScan Open-Source Raspberry Pi based 3D-Scanner*. 2020-09. Available also from: https://fabscan.org/.
4. *The Free 3D Printable Laser Scanning System For the Raspberry Pi*. [N.d.]. Available also from: http://www.freelss.org/.
5. ZUZA, Mikolas. *Fotogrammetrie - 3D skenování s použitím fotoaparátu či mobilu*. 2018-03. Available also from: https://blog.prusaprinters.org/cs/fotogrammetrie-3d-skenovani-s-pouzitim-fotoaparatu-ci-mobilu_7811/.
6. THINGIVERSE.COM. *The $30 3D scanner V7 updates*. 2016-09. Available also from: https://www.thingiverse.com/thing:1762299.
7. THINGIVERSE.COM. *OpenScan - 3D Scanner v2 by OpenScan*. 2018-08. Available also from: https://www.thingiverse.com/thing:3050437.
8. *OpenScan Cloud*. [N.d.]. Available also from: https://en.openscan.eu/openscan-cloud.
9. *Node-RED*. [N.d.]. Available also from: https://nodered.org/.