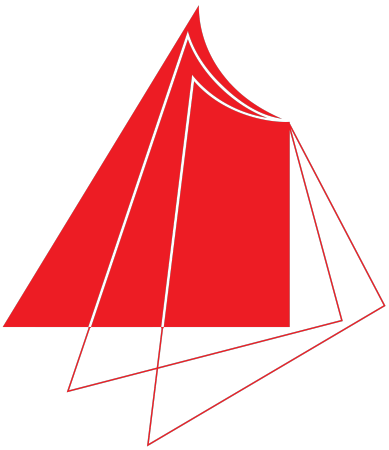

HOCHSCHULE KARLSRUHE - TECHNIK UND WIRTSCHAFT

Development HSkAnner3D

Benedikt Reinberger #62484

Contents

1 What is HSkAnner3D?

Figure 1: A similar physical approach to this one by MakeShape [**?**] will be taken for the array

HSkAnner3D is

- a 3D scanner → creates 3D models from 3D objects placed in the scanning space - the goal is to be able to do full body scans of humans
- a local system → no external dependencies, just mains power
- easy to use → can be operated by just about everyone

It uses

- an array of lights, controlled by lighting controllers → to illuminate the subject
- an array of cameras, each controlled by a single board computer (SBC) → to capture 2D color images of the subject from many different angles
- a powerful computer → to calculate 3D models from a set of 2D images using photogrammetry software and for controlling the arrays
- network cables and a network switch → to connect the SBCs and lighting controllers to the computer

2 System Architecture

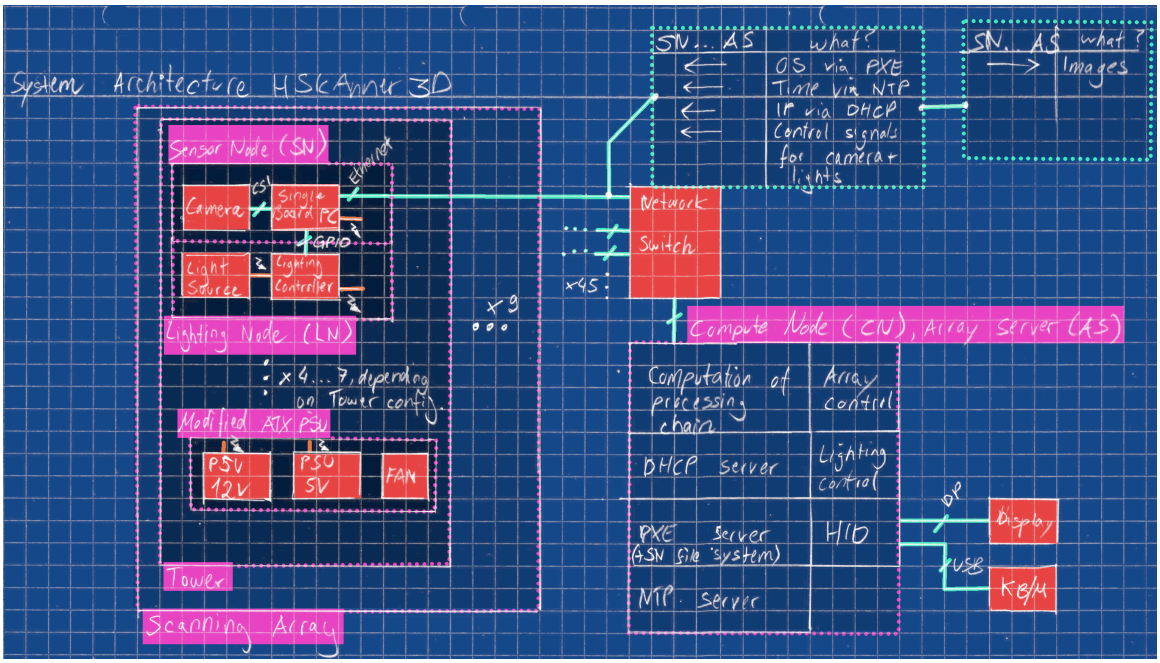

Figure 2: Preliminary architecture of the scanning system.
Legend
Pink Building block
Red Hardware component
Cyan Data (line)
Orange DC power

Note: I am aware that there are adapters to use two cameras with one Raspberry Pi. But I have done some research and it seems like the reliablity of those is not very high, thus they are not suitable for this project.

3 Why is it necessary to build HSkAnner3D?

Hochschule Karlsruhe does currently not have a 3D scanner, so it's missing out on the core features of such a system:

- being able to recreate complex shapes in 3D space without much effort
- creating 3D video data for use with e.g. motion tracking (will only be implemented if I have the time)

There is a plethora of arguments for building your own scanning system over buying a finished product:

- Cost
  Commercial solutions like the one MakeShape is offering cost large amounts of money. Their scanner uses 100 cameras, but costs around 43000€. HSkAnner3D has a total

project budget of 5000€(including research & development cost). Cost is also the reason why HSkAnner3D uses photogrammetry as a technique to generate 3D data, since depth cameras are very expensive.

- Flexibility
  Since all parts of the system, be it hardware or software, are designed to be modular, interchangable and adjustable, the system can be altered at will to suit different needs in the future.

- Easy maintenance
  The modularity helps make the system be more easily maintainable. The hardware and software is written and assembled with repairabilty in mind.

- Well documented
  As the project is developed in-house, there's no secret sauce. All code and design assets are available for future improvements and enhancements by students of the Hochschule.

4 How does HSkAnner3D work? The operation principle of the system

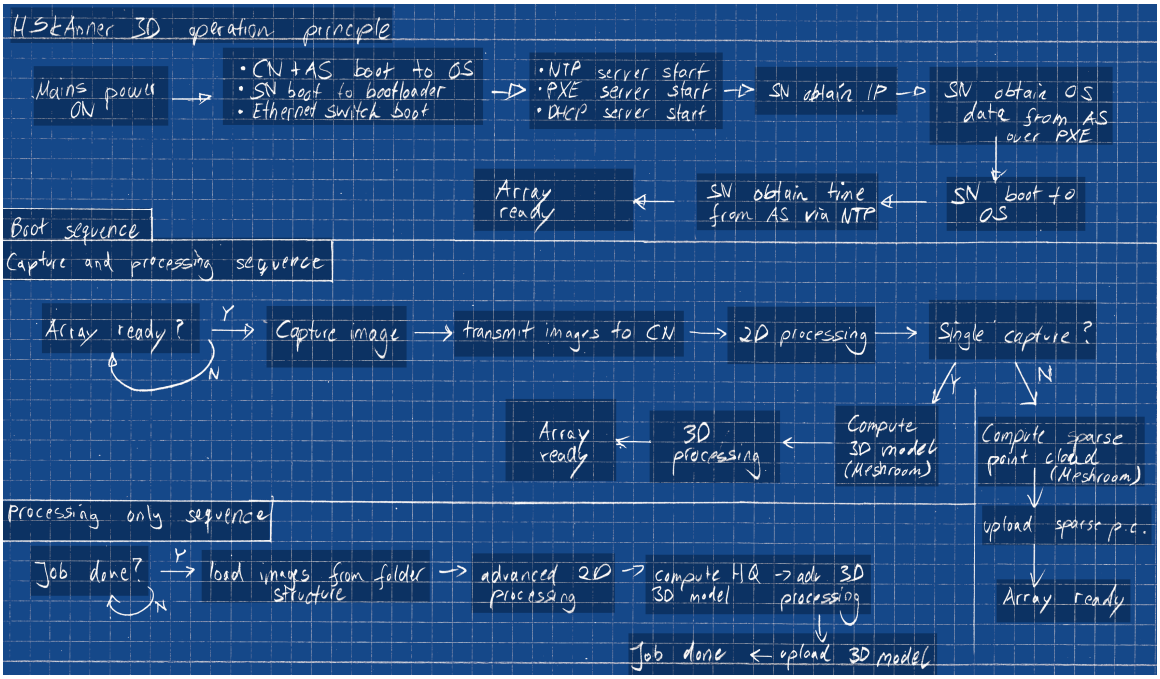

Figure 3: HSkAnner3D working principle flowchart

5 How are we going to archive the goal?

5.1 Introduction, Overview

To get to our desired goal to have a working scanner that works reliably with a slick GUI, we have to break it down into many smaller goals. Since the scanner will be used to promote the Hochschule, there's some additional steps to cover. The timeline is as follows:

- Build a small rig with 9 cameras to write software and set up the development side of things
- Set up all the required servers and development environment:
- Set up and automate the (capture and) processing chain/sequence
- Design the full scanning array with all 45 SN and lighting
- Talk to the workshop people to ensure good design
- Build one tower of the final scanning array,
- Calibrate tower (focus, (photogrammetry) settings)
- Write software to interface with the lighting system in the tower
- Build the remaining towers
- Align and calibrate the remaining towers
- Write a GUI to interface with the previous automation scripts or take over the functionality of those entirely. At this point, the scanner is a fully working system
- Write additional documentation and Praxissemesterbericht
- Render promotional video and art for presentation of the scanner
- Work on additional features

5.2 Software development rig

To set up the development environment as well as installing and configuring the required software, a development setup will be built. It consists of

- 9 sensor nodes (SN), each consisting of a Raspberry Pi 4B as a network controller and a camera module V2, with a 8MP $1/4$" sensor and $79d$ FOV $\mathrm{f}/2$ lens (power is supplied with PoE)
- one gigabit network switch, PoE capable to connect the SN with the CN and AS
- one compute node (CN) and one array server (AS): AS for controlling the SN and getting the images, CN for calculation of the 3D model from the 2D data - In this setup, CN and AS share the same hardware

5.3 Setting up the required software

For the scanner to function as such, software has to be set up.

- enabling netboot (PXE) on the SNs by flashing a new bootloader - booting into the SN using an SD card that has the necessary bootloader and a handy script for flashing it
- DHCP server on the array server (AS) to assign an IP address to every sensor node when connected to the local network - the built-in DHCP server of DNSMASQ will be used
- DHCP client configuration on the SNs - no modification was needed for the Raspberry Pies using Raspberry Pi OS, since the default is to use the IP from DHCP
- PXE server and file system on the AS to boot all the SNs from the network, omitting the SD card - the server will be set up using DNSMASQ
- NTP server on the AS so that all the SNs share the same time as the AS - this is used for syncronized capture
- NTP client configuration on the SNs to get the time from a local time server and not from the web since the SN are not connected to it for security reasons
- Compound Pi (CPI) client on the AS to control the cameras on the SN and download the images to the CN - since AS and CN share hardware, the images don't have to be moved from the AS to the CN
- CPI server on the SNs to be able to be controlled remotely
- enabling SSH on the SN for development and debugging purposes

5.4 Automation of the processing chain

The "Capture and processing sequence", as seen in figure 3, has to be automated, to later enable operators to use the device "with the push of a single button". The automation is split into subsections, to make maintenance and modifications easier. The subsections consist of

- taking all the photos by executing a single script/command
- processing photos (built into the chain, but not yet used. Will only be implemented if needed) by executing a single script/command
- generating 3D data from photos by executing a single script/command
- processing 3D data (same as with photo processing, no 3D processing is done outside of the meshing software, Meshroom) by executing a single script/command
- all of the above, but with one command in total

5.5 Design of the final scanning array with all 45 cameras and lighting

Once the software has been set up for the basic operation using a single command, the final scanning array can be designed. There are a number of substeps that go into the design process:

- find out where to place the cameras
- find out how many lights are required and where to put them
- find out whether LED striplights or spots are more suitable for the job
- design a physical construction concept that is
  - cheap (after the development setup build, we have 3275.13€left from our 5000€budget. We need 35€more SN at a cost of 52.61€each, and we want to build 9 towers, so the total budget per tower is 160.53€, or about 150€with an included buffer)
  - easy to manufacture
  - solid in the long term (no deformation or degradation that jeopardizes the system integrity)
  - easy to adjust
  - easy to maintain
  - easy to (dis-)assemble once built (array disassembly, for transport)
  - portable when split into its modules

5.6 Ensuring good physical design

Since mechanical engineering is not my field of expertise, the design of the physical structures will be talked trough with the workshop people to find the correct

- material choice
- design for manufacturability
- design for CNC machining

5.7 Building one tower

After the design has been finalized, parts for one tower will be ordered and it will be assembled. This is done to iron out any mistakes that were overlooked on the hardware or software side, so that the remaining towers can be as good as possible.

5.8 Single tower calibration

When the tower has been finished, it will be set up for use and tested. If it passes the test - meaning all cameras align in the photogrammetry software when scanning a human - we can move on. If not, the design will be modified to combat possible shortcomings.

5.9 Lighting control from the AS

Once the tower has passed the test, it is time to set up lighting control over the network. This will again be done in steps:

- control the lights using the Raspberry Pi GPIO from the Pi directly
- implement functions for fading the lighting in and out as well as RGB effects
- set up a (Python?) server on the Pi that is able to control the GPIO and as such can control the lights using the functions or directly writing values
- set up a (Python?) client on the AS to send lighting requests to the server on the Pi
- integrate lighting control in the automation process

5.10 Building of remaining towers

Now that the towers can be fully controlled, it's time to build the remaining towers and order the rest of the cameras needed.

5.11 Calibration of towers

All lenses have to be focused to the middle of the scanner. To do that easily, some setup is required:

- boot the Pies to desktop OS
- set up to automatically log in and display camera view once booted
- connect an head mounted display (HMD) to HDMI port
- focus all cameras manually to central object

Once the focus is set, the software settings have to be adjusted to the array:

- camera ISO, shutter speed
- lighting brightness and fading duration
- photogrammetry setting to reliably align the images and get a good quality model as fast as possible

5.12 GUI for scanner control

At this point, all aspects of the scanner have been automated, but the interface is command-line only. A graphical user interface (GUI) is more intuitive for most users and can give quick access to all the required settings and dials of the scanner. It will be created either using Qt (C++) or Tkinter (Python).

5.13 Documentation and Praxissemesterbericht

Thus, the scanner as a complete system has been finished. It is possible to change settings from a GUI and a scan can be made with the click of a single button. This marks the start of the documentation and Praxissemesterbericht phase.

5.14 Promotional material

Since the scanner will be used by the Hochschule not only internally but as an attraction on presentation days for the public to see, promotional material will support the concept and add to the experience. I have two concrete ideas:

- a logo for the scanner
- a video that will loop on a display near the scanner to show the process from capture to 3D model

5.15 Additional features

To increase the value gained from the setup, additional software features can be developed:

- $360d$ video
- motion capture using said video
- rendering the 3D model into a scene using e.g. Blender (Mt. Rushmore, ...)
- video render of 3D model for viewing pleasure
- making the 3D models available from a browser (writing a website, ensuring privacy protection)
- scanning and meshing improvements, e.g.
  - using structured light [**?**]
  - using polarized light
  - increasing the number of cameras
  - taking muliple photos in quick succession to increase the effective camera resolution