Yannic Schröder
schroeder@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Alexander Scholz
scholz@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Kai Berger
berger@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Kai Ruhl
ruhl@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Dr. rer. nat. Stefan Guthe
guthe@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Prof. Dr.-Ing. Marcus Magnor
magnor@cg.tu-bs.de
Computer Graphics Lab, TU Braunschweig

Multiple Kinect Studies

Technical Report 2011-09-15

October 5, 2011

Computer Graphics Lab, TU Braunschweig

Zusammenfassung

Die Microsoft Kinect ist ein neuartiges Aufnahmegerät für Farb- und Tiefenbilder (RGB-D), das für die Unterhaltung entwickelt wurde. Auf Grund der besonderen Leistungsfähigkeit und des geringen Preises im Vergleich zu üblichen Aufnahmegeräten für Tiefeninformationen ist die Kinect Thema einer Vielzahl wissenschaftlicher Untersuchungen.

Wir untersuchen, welche Möglichkeiten sich bieten mehrere Kinects in der gleichen Umgebung zur Aufnahme von opaken und durchsichtigen Objekten zu verwenden. Dabei entstehen Probleme durch Laserinterferenzen die durch schnell drehende Scheiben gelöst werden, indem ein Zeitmultiplex erzeugt wird. Außerdem stellen wir einen Algorithmus zur Qualitätsauswertung von Tiefenbildern vor und vergleichen die Bildqualität von Bildern die ohne Multiplex aufgenommen wurden mit jenen, die unser Multiplexsystem erzeugt hat.

Abstract

The Microsoft Kinect is a new color-depth (RGB-D) tracking device designed for home entertainment. With its great performance and low price compared to state-of-the-art depth tracking systems it is the target of a variety of academic research.

We investigate the possibilities of using multiple Kinects in the same environment for capturing opaque and translucent objects in motion. Arising issues with the laser subsystems interfering with each other are solved using fast rotating disks, creating a time division multiple access (TDMA) scenario. We also introduce an algorithm to evaluate the quality of captured depth images and compare the quality of depth images created with our multiplexing technique to depth images obtained without multiplexing.

Contents

Chapter 1

Introduction

The Microsoft Kinect's [Mic11a] regular purpose is home entertainment. It is an accessory for the Microsoft Xbox 360 [Mic11c] gaming console and used as a new input device that replaces the standard game pad. Therefore, Microsoft delivered a quite robust depth tracking system. It is designed as a stand-alone solution with only a single device in the present. Thus, running multiple Kinects simultaneously may lead to various unexpected errors which are inherent to the design.

The focus of the technical report is therefore the evaluation of quality degradation in depth images for scene setups with multiple Kinects. The degradation can be explained by examining the Kinects laser subsystem. Each Kinect has one infrared laser that is passing through a special lens which causes a unique speckle pattern to be projected onto the surfaces that the Kinect is facing. The reflected pattern is detected by an infrared camera and converted into a depth image. Thus, the computer or gaming console to which the Kinect is connected does not need to compute the depth image, saving extra CPU time. When multiple Kinects are facing the same surface, each of them projects its own laser pattern onto it. Each infrared camera now detects the overlay of all speckle patterns and the Kinect is unable to distinguish the dots from its own pattern from the ones projected by other Kinects. Therefore, the quality of the depth images degrades with the number of Kinects concurrently running. The loss in quality is dependant on the designated setup. Consider the following capturing examples: Four Kinects are placed in the corners of a room and an actor stands in between them. The depth image quality is reduced because each Kinect captures the interference of two other Kinects. In a setup with four Kinects facing a single wall the quality of the depth images degrades to an untolerable point. This is caused by the massive interference of the four lasers, see Fig. 1.1 for a comparison.

Obviously there is a need to improve the image quality. This can be done by time division multiple access (TDMA) of the Kinects. Each Kinect

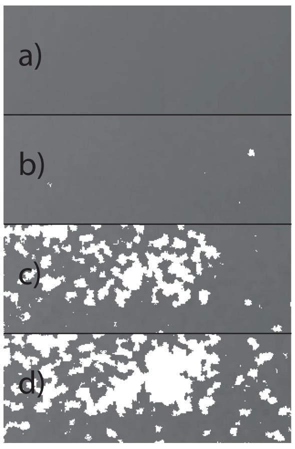

Figure 1.1: **The image quality degradation with different numbers of Kinects facing the same surface.** a) 1 Kinect, b) 2 Kinects, c) 3 Kinects, d) 4 Kinects. Each stripe is from the same area of the image of one of the Kinects.

gets a well defined time slot in which it can project its own pattern and capture the result without any interference. We introduce a system to solve this problem.

Chapter 2

Time division multiple access approaches

As explained before, the Kinects laser is the crucial point in multiplexing. In this chapter we will discuss some approaches.

2.1 Toggling the laser diode

The simplest solution for multiplexing is to toggle the laser diode itself. However, until now it is impossible to do this by software. Microsoft neither disclosed the needed specification for it nor shipped an API to do so. Besides, the Microsoft Kinect SDK [Mic11b] doesn't allow to toggle the laser. So, it is not known if toggling the laser by software is possible at all. The other option would be to toggle the laser with a special hardware switch. This however would require invasive modifications as shown by iFixit [iFi10].

2.2 LCD shutter

A different approach is to use an LC-Display with a single big pixel as a shutter for the laser similar to shutter-glasses for 3D applications [Ama]. We tested this approach with a microcontroller and an LC-Display taken from shutter-glasses. However, since the LCD-Shutter is only able to block a certain amount of the lasers light, see Fig. 2.1, it did not lead to the intended result.

2.3 Laser shutter for optical tables

For experiments with lasers in general there are various devices to be mounted on optical tables that are able to shutter lasers precisely and very fast. The biggest disadvantage is the price of these units. One of these shutters costs

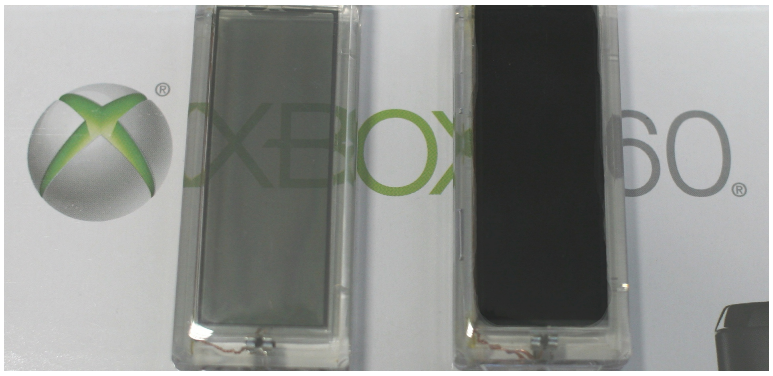

Figure 2.1: **Light blocking with LCDs.** Left: unpowered and transparent, right: powered and opaque. Although the powered LCD [Ama] appears opaque the intensity of the infrared laser is not degraded enough for blocking.

about five times more than a Kinect. Furthermore these shutters are designed for a single laser beam whereas the Kinect emits a cone of laser beams that might not pass correctly.

2.4 Hard disk drive voice-coil actuator

Maguire et al. [MSS04] show how to use an actuator from an old hard drive to build a shutter. This was not implemented for Kinects because vibrations would become a major issue of this setup.

2.5 Revolving disk

The idea that was implemented for this technical report is a fast revolving disk with a gap. The laser is blocked by this disk except for the time when the gap is is allowing the laser to project its pattern into the scene. Each Kinect is mounted to such a disk rotating at the same speed but with a different phase, ensuring only one laser can project at any given time.

The disks have to be precisely synchronized as it is intolerable if timing varies due to wrong disk speed. Therefore, we use stepper motors as they can be precisely controlled by changing the currents in their windings to do exactly one step in a defined time. Using the amount of steps needed for a complete revolution it is easy to rotate the disk at the required speed. Furthermore, it is possible to synchronize the motors by synchronizing the currents in their windings.

This way we get a highly reliable setup with disks blocking the laser completely and stepper motors controlling the exact speed of the disks.

Chapter 3

Related work

There are several publications dealing with multiple Kinects or laser shuttering in general but none of them combined both subjects.

A system for blocking laser beams with a moveable part inside the laser path is introduced by Ichinokawa [Ich92].

Such a system is implemented by Maguire et al. [MSS04]. They use voice-coil actuators from hard disk drives to build shutters for optical tables. They investigated the usability of hard disk drives for this use in general, but as the Kinect is a camera system and even small impacts or vibrations as introduced by the actuators may harm the image quality by blurring the image, these actuators are not suitable for our application.

Wilson et al. [WB10] combine multiple depth cameras that are predecessors of the Kinect with the same underlying technique in a setup to capture a bigger space than is possible with one device. They do not cover the problem of laser interference and overlap the projections only as much as needed for covering the scene without gaps.

Furthermore, Takeuchi et al. [TNNH11] build an interactive motion area out of four Kinects with big overlapping areas but omit the interference caused by this setup completely.

Recently Maimone et al. [MF11] notice arising problems with interference of the laser subsystems and apply a hole filling algorithm in a post processing stage which is able to fill small holes by assuming planar surfaces. It fills the holes to close these surfaces. This approach has the following limitation: holes in the projected boundary of captured objects may be filled, so that the object boundary information is wrongly reconstructed. This effect can already be noticed in the hole filling procedure implemented in the Kinect itself [Ope11].

Chapter 4

Implementation

4.1 The revolving disk

4.1.1 Use case

Note, that we implemented a use case for four Kinects concurrently running. However, our system can be scaled to any other number of Kinects by designing a disk (see below) with the correct number of segments.

4.1.2 Electrical components

The hardware consists of four main parts as seen in Figure 4.1. The computer is able to send commands to the microcontroller and receive status information via USB. The microcontroller processes commands of the computer and generates the signals for the stepper controllers. These stepper controllers process the commands to ensure that each magnet coil in the stepper motors gets the correct current to do the next step. The stepper drivers are plugged into the circuit boards. They generate the currents for the solenoids. The stepper motors are connected to the drivers with four wires each resulting in a bipolar motor setup. Note, that the steppers have additional wiring for a unipolar setup which is not used here. The system consists of standard hardware available at every electronic components supplier.

Microcontroller circuitry

The microcontroller is the central point in the setup. For testing purposes it is mounted on a solderless breadboard [Rei]. However a soldered circuit board would be preferred for frequent uses. The microcontroller is an Atmel ATmega48 [Atm11], but every other Atmel microcontroller which is compatible with the V-USB [Obj11] library can be used as well as the other requirements for the setup are very low. The ATmega48 runs at a clock

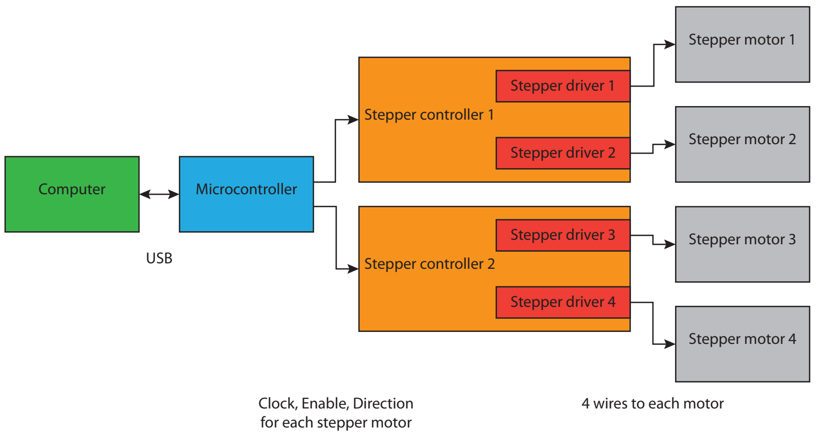

Figure 4.1: **The block diagram of the electrical components.** The computer is connected via USB to the microcontroller which allows bi-directional communication. The microcontroller sends commands as logic high and low levels to the stepper controllers which generate currents accordingly for the stepper motors.

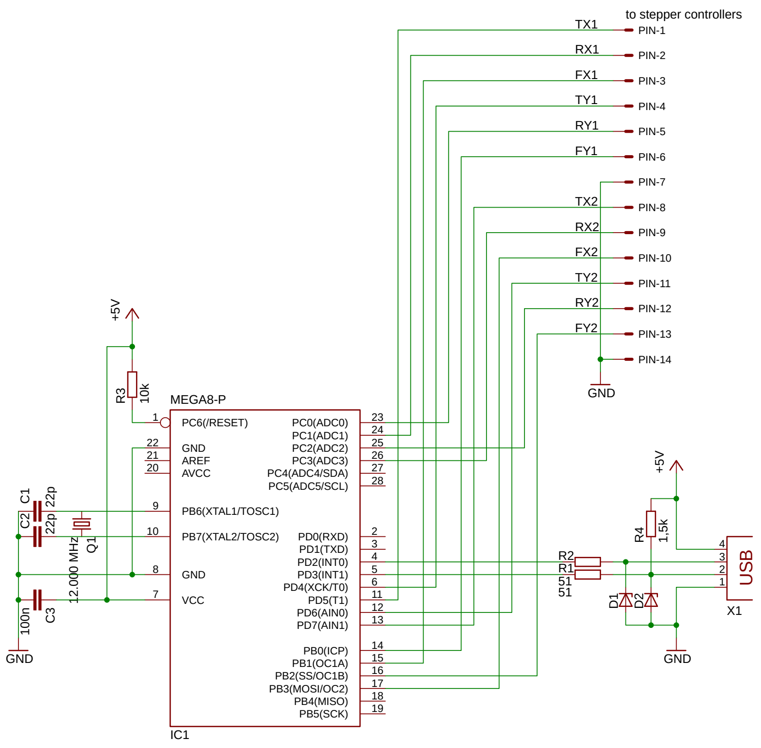

Figure 4.2: **The electric circuit of the microcontroller board.** See Figure 4.1, blue box.

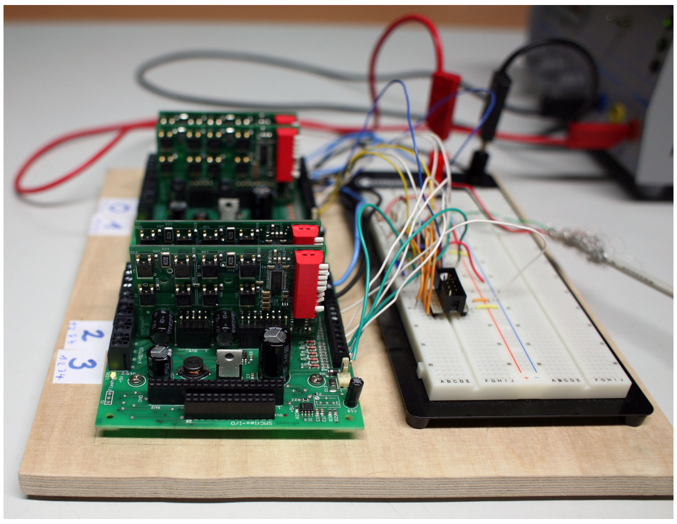

Figure 4.3: **The circuitry board.** The microcontroller breadboard and the two stepper controllers are mounted on a board. The four stepper drivers are plugged into the stepper controllers. The output ports for the motors are labeled 0 to 3.

speed of $12MHz$ as this is one of the possible clock speeds for V-USB. The circuitry to the left of the ATmega48 in Fig. 4.2 is implemented as proposed by Objective Development [Obj11]. The circuitry for USB to the right of the ATmega48 is a possibility differing from the reference implementation of Objective Development and proposed by Krude [Kru11]. This allows the ATmega48 to be powered by $5V$ through the USB as the reference implementation uses $3.3V$ powering. This is essential because our stepper controllers need a high level of at least $5V$. Fig. 4.2 shows further the connections of the ATmega48 to the stepper controllers. TX1, TY1, TX2 and TY2 are the clock signals for the steppers. On each rising edge the corresponding stepper moves one step. RX1, RY1, RX2 and RY2 define the directions of the steppers where a low level is one direction and a high level the other. The exact direction (left/right) depends on the wiring between the driver and the stepper. FX1, FY1, FX2 and FY2 are the enable signals for clearance. A high level results in powering the stepper. Without clearance the stepper is without any current and can be easily moved by hand. With clearance this gets harder as the solenoids receive power and hold the axis in its position. Obviously, it is necessary to set clearance as the motors are unable to move without it.

Stepper controller and driver

The stepper controller and driver are ready-to-use circuitry. It is a modular design so the driver can be changed with the motor to power. Please note,

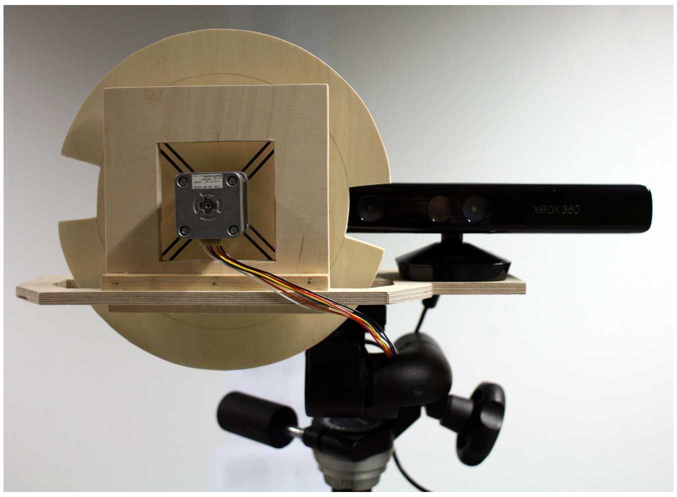

Figure 4.4: **The revolving disk platform** is mounted on a tripod with the Kinect standing at a defined position on the platform. The stepper motor holding the disk is suspended with rubber rings to minimize vibrations. This image show the disks state, where the Kinect is permitted to emit laser light into the scene.

that in our setup the stepper motors do not need much torque as our disks are very light weight. Therefore, the drivers only need to drive a maximum of $1A$. The drivers are configured to the needed currents for the motors and to use half-step mode. The half-step mode reduces resonant noise from the setup and increases the maximum torque of the motors. This results in steeper ramps for changing motor speed.

Stepper motor

The stepper motors used in our setup are the smallest ones available. As mentioned above there is no need for high torque, so small and inexpensive motors are feasible.

4.1.3 Mounting platform

The revolving disk platform is the part of the system containing the stepper motor and holding the Kinect in a defined place to be able to cover the laser with the disk, see Fig. 4.4. It is mounted on a tripod with a snap closure for easy exchange of the camera. The stepper motor is held in place with four rubber rings to ensure that any imbalance in the fast rotating disk will not influence the Kinect and avoids blurry images caused by vibration. The disk is attached to the axis of the stepper motor with focus on minimal imbalance. The platform is designed to fit the dimensions of the Kinect with a slot in the disk as wide as the lasers optic and a position for the Kinect allowing to only cover the laser with the disk and leaving both cameras unaffected. Each Kinect is placed on its own platform resulting in four platforms when using four Kinects.

4.1.4 Rotating disk

The disk is designed for setups with four Kinects. Each slot has an opening angle of 22.5°. This solution is derived as follows: The disk shall be used with four Kinects so the disk must have four defined states. Three with the laser blocked and one with the laser projecting. As the disk needs to rotate very fast a single slot in the disk would lead to undesired imbalance and vibration. Therefore, the disk has eight states with two slots opposite to each other, see Fig. 4.4. The disk divided by eight leads to sections with an angle of 45°. To ensure that only one laser projects at a time the slot for each Kinect is narrowed farther. 12.5° of the projection slot are blocked; 22.5° are open, followed by another 12.5° blocked. Without these additional blocks the lasers of two adjacent Kinects would have a time instant with one shutter closing and the next opening leading to both lasers projecting at the same time. Using this design, the laser emitter of the last Kinect is definitely closed when the next is opening.

For other numbers of Kinects the disk has to be changed accordingly. For each Kinect the disk should have one segment and another at the opposite side for balance.

4.1.5 Rotating disks speed

As mentioned earlier the disk needs to rotate at a well defined speed to align with the camera images. The Kinect cameras run at $30fps$ with a rolling shutter. As we are not able to control the time instances of the image acquisition we cannot synchronize to an exact frame. The shutter will always open and close somewhere in the middle of a frame resulting in a half frame exposed by the laser and therefore in half a depth image. To be able to get a complete depth image we have to take two subsequent frames and stitch them together later. With four Kinects we have two images exposed by the laser followed by six blank images exposed by the other Kinects lasers, see Fig. 4.5. The shutter shall be opened every eight frames the Kinect acquires. Therefore, a half revolution has to be finished in $\frac{8}{30}s$. Thus, a complete revolution has to be finished in $\frac{16}{30}s$. With the stepper motors in half step mode each revolution consists of 400 steps requiring 400 clock cycles to step through. 400 steps have to be finished in $\frac{16}{30}s$ leading to $750\frac{steps}{s}$. In general this can be expressed by:

$$c = \frac{s \cdot f}{4 \cdot k} \tag{4.1}$$

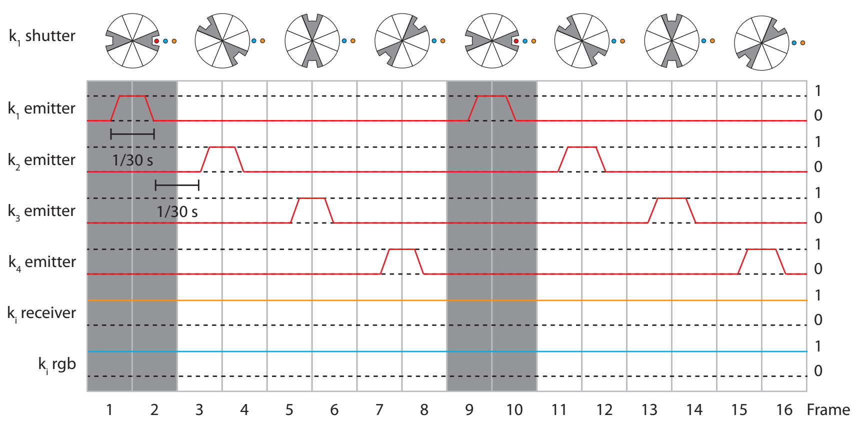

Figure 4.5: **The multiplex of four Kinects with time instants for laser exposition.** Each emitter is visible $\frac{1}{30}s$ while the receivers and rgb-sensors are open at all times. Note disc turning 45° or one segment over 2 frames. Reproduced from [BRB+11].

With c being the clock speed in Hz for the stepper motors, s the number of steps per revolution (normally 200 steps in full step mode, 400 steps in half step mode, see stepper motor data sheet for details), f the frames per second (30 frames here, as the Kinect always delivers $30fps$ for the depth stream) and k the number of Kinects to multiplex.

4.1.6 Exploiting the Kinects laser security system

The Kinect contains a security system for the infrared laser diode. The laser is quite strong as it is required to be visible even on surfaces far away from the system. Therefore it is hazardous to the human eye to stare directly into the laser. The security system works as follows: When the integrated circuit recognizes that only a small part or nothing of the speckle pattern is visible in the infrared image it is likely that something is blocking the laser diode. Hence it assumes it may be a human head and turns off the laser to prevent eye damage. As it is only able to check if the occluder disappeared by checking the speckle pattern again it turns the laser on after a few seconds and rapidly off again if the occluder is still in front of the laser. With our disk being such an occluder and confusing the security system we need a way to disable this security feature.

As found by the OpenKinect project [Ope11] it is possible to disable the laser security. Via a control command send to the Kinects control endpoint it is possible to set the parameter 0x0105. If the value of this parameter is nonzero the security is enabled, if this parameter is zero it is disabled. This worked for our multiplexing system. We also found the security system is too slow to detect very short occlusion of the laser.

4.1.7 Control software

The ATmega48 controls the stepper motors, but it needs commands to do this. To be able to use the multiplex system with elaborated Kinect capturing software we decided to connect the microcontroller via USB to a computer. This way we are able to integrate the control of the disks into any kind of software. As a proof of concept we implemented a command line tool which is able to send commands via USB to the ATmega48. These commands include enabling the steppers, setting a step cycle duration, switching the motor's direction, resetting the internal step counter for calibration and rotate the disks in a way that each Kinect is behind a different section of it. Further we added a status command to get information from the ATmega48. This information includes the current step cycle duration, the enabled motors and their direction.

4.2 Image stitching

After obtaining images from the Kinect the half images need to be stitched. Our algorithm analyses the stream of depth images, finds images that belong together and stitches them as follows:

For each pixel it counts the number of images with valid pixel data. If the result is zero, the pixel in the result is a measurement error (white). If the result is one, the pixel is set to this value. If the result is greater than one, it averages the values over the images to reduce noise in this part of the image as a side effect. This is optional, as it would suffice to use only one of the valid values. The process is visualized in Fig. 4.6. In part a) the red tinted pixels are from the first half image, the blue tinted pixels are from the second half image, pixels without valid values in both images are white, pixels with valid values in both images are tinted purple. Part b) shows the final result as generated by our algorithm.

In scenarios with fast moving objects being captured this stitching may lead to ghost images due to the averaging of the half images. This ghosting can be eliminated by taking only one half image into account for each pixel. It is advisable to set a priority, e.g. if the pixel is not a measurement error in the first half image always take that pixel information and only if it is not available take the data from the other half image.

4.3 Loss of framerate

As seen in Fig. 4.5 each Kinect is only able to acquire two depth images out of eight in a four Kinect TDMA setup as the laser is only exposed in the time slot. Further those two images are stitched together forming one full image. This leads to a framerate reduction to $\frac{1}{8}$ or 12.5% of the original

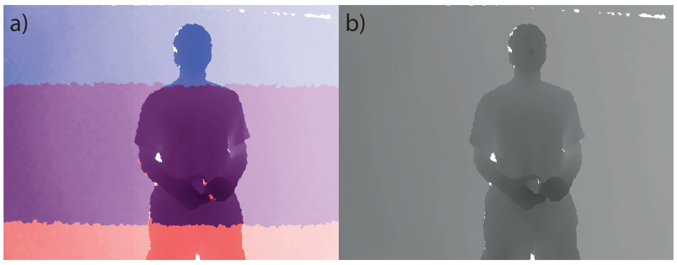

Figure 4.6: **Visualization of the stitching process.** a) the overlay of the first half image (red) and the second half image (blue) with parts covered by both images (purple), b) stitched result of both half images

framerate. By the design of our system we do not block the RGB-sensor at all, leading to the full framerate captured here. If the framerate of the depth image stream is crucial in the designated setup, it is advisable not to let each Kinect project its pattern alone but to do a 2-by-2 multiplex with two Kinects projecting at the same time. This will double the framerate with a moderate loss in depth image quality.

4.4 Calibrating the RGB-D system

As introduced by Berger et al. [BRB$^+$11] the system is calibrated using a mirroring checkerboard, see Fig. 4.7. This checkerboard pattern is visible in the RGB image as well as in the depth image. The mirroring patches will reflect the laser to infinity resulting in deflection holes in the depth image and reflect visible light to another surface in the RGB image resulting in a different color and brightness. The white diffuse patches reflect the laser of the Kinect as expected and therefore give valid depth values. In the RGB image the white diffuse patches are visible as well. This way it is possible to calibrate the system with the Matlab calibration toolbox [Bou10].

4.5 Evaluation of depth image degradations

There are several aspects which lead to a general loss of Kinect's depth data quality. One aspect is the surface reflectance of an object in the scene, which causes the infrared light to be reflected to directions other than to the infrared camera. While general noise also reduces the depth accuracy, additional infrared light (e.g. daylight) can strongly harm the depth image.

As the Kinect is designed to detect only its own individual speckle pattern in the infrared image, additional speckle patterns do also harm the depth image of a Kinect. In general the Kinect produces a depth map of 11-bit values. If the depth can not be calculated with a certain confidence

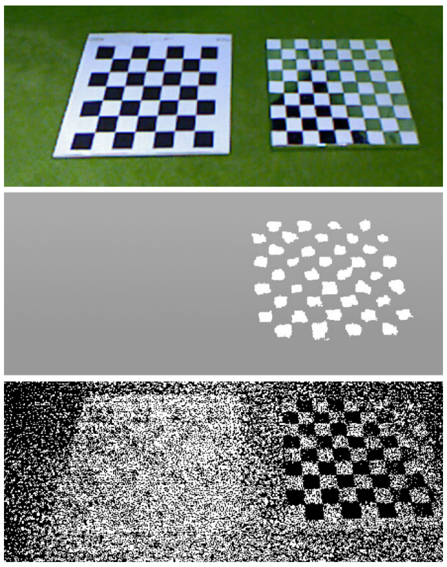

Figure 4.7: **Comparison of standard checkerboard versus mirroring checkerboard.** A standard checkerboard is not visible in the depth image. The mirroring checkerboard is visible in RGB and depth images. The bottom row shows the raw data from the infrared camera (thresholded for better visibility). Reproduced from [BRB+11].

value in one region, the missing data is set to the maximum 11-bit value 2047 marking a calculation error. This way, we discovered that therefore unusable parts of the depth image increase with the number of Kinects pointing into the same scene. In other words: the more interfering infrared patterns in the scene, the higher the error.

We developed an algorithm to detect such errors and to rate the quality of the depth data. The following algorithm counts the errors in a scene. For this, a stream of $n = 100$ images of the same scene (i.e. recording time of three seconds) is loaded and the total amount of erroneous pixels is counted. This value divided by n gives the mean error count for the images and after a conversion to percent it acts as a representation of the depth image quality.

```
n = 100;                % use 100 images
IMAGES(640,480,n);      % prepare image data

% load n images
for i = 1:n
  IM = loadImage(n);
  IMAGES(:,:,n) = IM;
end

% count errors
err_count = 0;
for i = 1:640
  for j = 1:480
    for k = 1:n
      if IMAGES(i,j,k) == 2047
        err_count++;
      end
    end
  end
end

% compute error per image
err_im_count = err_count/n;
err_percent = (err_im_count/(640*480))*100;

output: err_percent;
```

Figure 4.8: **The error detection algorithm.** $n = 100$ images are loaded into the prepared data structure *IMAGES*, which leads to a space-time-volume with varying (by noise) image data along the 3rd dimension (time). Afterwards, all invalid pixels ($depth == 2047$) in all $n = 100$ images are counted. After dividing this total error-pixel count *err_count* by n we get the average error-pixel count per image (*err_im_count*). The final step converts this count to a percentage value, which is then returned.

Chapter 5

Results

We are able to improve the overall depth image quality greatly. The ground truth for the comparison is a stream of depth images captured without multiplexing and with four Kinects projecting its lasers into the scene. We compare these images to depth images taken with multiplexing the same four Kinects and a single laser projecting into the scene at any time. As clearly visible in Fig. 5.1, the percentage of errors (white pixels) decreases dramatically with multiplexing. The RGB image is unaffected by our multiplex system and always shows a valid image as the RGB camera is not blocked by the disk at any time. The Kinect is able to capture valid half images when the gap of the rotating disk is passing by and our stitching algorithm is able to reproduce full images out of the depth image stream. Afterwards, our algorithm for evaluating the image quality calculates the percentage of erroneous pixels for images obtained without multiplexing and for the stitched images. In scenes with objects only a rectangle around the object is taken into account for the calculation as this shows the influence of different materials. Fig. 5.1 shows that this is a good metric for depth image quality.

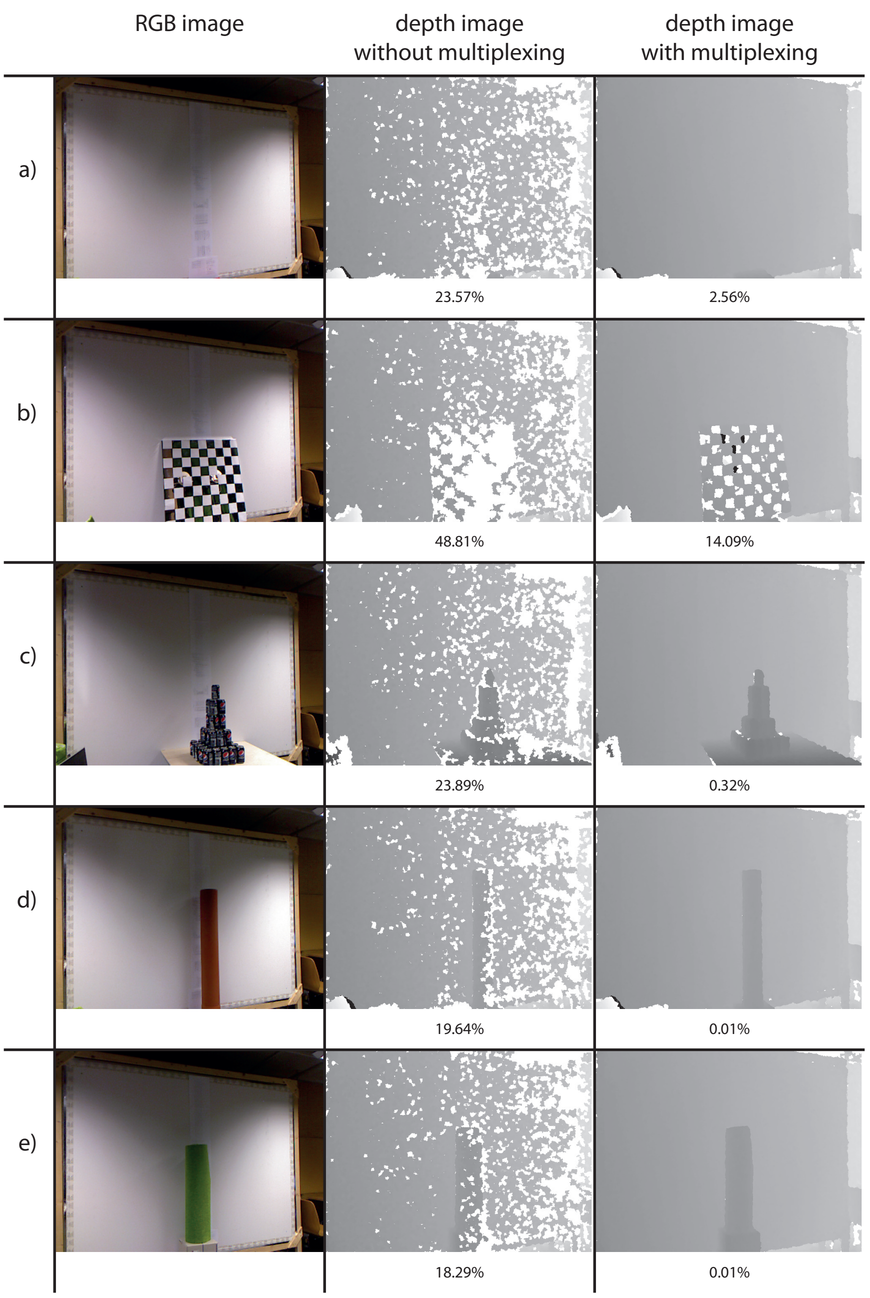

Figure 5.1: **Comparison of image acquisition with and without multiplexing for materials with varying specularity.** Percentages under the depth images show the fraction of erroneous (white) pixels. Note, that in scenes b) to e) only the part of the image with the object was taken into account for the calculation to show the effect of different materials. a) empty scene, b) mirroring checkerboard, c) specular pyramid of beverage cans, d) glossy plastic tube, e) diffuse carpet

Chapter 6

Conclusion

In this technical report, we introduced a system for time division multiple access of Kinects to a scene. The system is able to block the Kinects laser as this is the crucial point in muliplexing Kinects. We get an image stream from each Kinect with alternating valid half images and blank images and we are able to find the valid half images in this stream. Furthermore, we stitch these images together and form full images again. The side-effect is a reduced framerate in the depth images as each laser cannot access the scene all the time any more. While reducing the framerate the overall depth image quality increases greatly, see Fig. 5.1. The introduced system for multiplexing has been used for diverse capturing scenarios as proposed by Berger et al. [BRB$^+$11] [BRA$^+$11]. The overall image quality in scenarios like the ones introduced by Takeuchi et al. [TNNH11] and Maimone et al. [MF11] can be increased greatly.

6.1 Future work

In future work the influence of the disks rotating direction is to be investigated as it might interfere with the rolling shutter of the Kinect. Furthermore, we investigate the possibility of increasing the depth image quality by averaging subsequent depth frames.

Bibliography

[Ama] Amazon. Nvidia GeForce 3D Vision Glasses. http://www.amazon.de/exec/obidos/ASIN/b002ntp43y/.

[Atm11] Atmel Corporation. *8-bit Atmel Microcontroller with 4/8/16K Bytes In-System Programmable Flash ATmega48/V ATmega88/V ATmega168/V*, September 2011. http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf.

[Bou10] Jean-Yves Bouguet. *Camera calibration toolbox*, July 2010. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[BRA$^+$11] Kai Berger, Kai Ruhl, Mark Albers, Yannic Schröder, Alexander Scholz, Stefan Guthe, and Marcus Magnor. The capturing of turbulent gas flows using multiple kinects. In *Proc. CDC4CV 2011, to appear*, November 2011.

[BRB$^+$11] Kai Berger, Kai Ruhl, Christian Brümmer, Yannic Schröder, Alexander Scholz, and Marcus Magnor. Markerless motion capture using multiple color-depth sensors. In *Proc. Vision, Modeling and Visualization (VMV) 2011*, October 2011.

[Ich92] Kazuhiro Ichinokawa. Laser shutter mechanism. U.S. Patent #5,153,607, 1992.

[iFi10] iFixit. *Microsoft Kinect Teardown*, November 2010. http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/1.

[Kru11] Johannes Krude. *USB Relay*, September 2011. http://johannes.krude.de/projects/usb-Relay/.

[MF11] Andrew Maimone and Henry Fuchs. Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *ISMAR 2011, to appear*, 2011.

[Mic11a] Microsoft Corporation. Microsoft Kinect. http://www.xbox.com/en-US/kinect, September 2011.

[Mic11b] Microsoft Corporation. *Microsoft Kinect SDK for Windows*, September 2011. http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/.

[Mic11c] Microsoft Corporation. Microsoft Xbox 360. http://www.xbox.com, September 2011.

[MSS04] L. P. Maguire, S. Szilagyi, and R. E. Scholten. High performance laser shutter using a hard disk drive voice-coil actuator. *Review of Scientific Instruments*, 75(9), September 2004.

[Obj11] Objective Development Software GmbH. *Virtual USB port for AVR microcontrollers*, September 2011. http://www.obdev.at/products/vusb/index.html.

[Ope11] OpenKinect project. *Protocol Documentation*, September 2011. http://openkinect.org/wiki/Protocol_Documentation.

[Rei] Reichelt Elektronik. STECKBOARD 2K1V :: Experimentier-Steckboard 1280/100 Kontakte. http://www.reichelt.de/?ARTICLE=67679;PROVID=1024.

[TNNH11] Toshiki Takeuchi, Totaro Nakashima, Kunihiro Nishimura, and Michitaka Hirose. Prima: Parallel reality-based interactive motion area. In *SIGGRAPH 2011*, August 2011.

[WB10] Andrew D. Wilson and Hrvoje Benko. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, pages 273–282, New York, NY, USA, 2010. ACM.