

上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

学士学位论文

THESIS OF BACHELOR



Project Title: Control of Virtual and Real Manufacturing
Systems via AR/VR techniques

Name: <u>Zhiqi Chen</u>	Student ID: <u>516370910245</u>	Major: <u>ECE</u>
Name: <u>Tianyi Ge</u>	Student ID: <u>516370910168</u>	Major: <u>ECE</u>
Name: <u>Yue Wu</u>	Student ID: <u>516370910151</u>	Major: <u>ECE</u>
Name: <u>Yue Zhang</u>	Student ID: <u>516370910055</u>	Major: <u>ECE</u>
Name: <u>Junwei Deng</u>	Student ID: <u>516370910085</u>	Major: <u>ECE</u>

Sponsor:	<u>Professor Mian Li</u>
Institute:	<u>UM-SJTU Joint Institute</u>
Semester:	<u>2019-2020 Summer</u>

上海交通大学

毕业设计（论文）学术诚信声明

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日期： 年 月 日

上海交通大学

毕业设计（论文）版权使用授权书

本毕业设计（论文）作者同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本毕业设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本毕业设计（论文）。

保密☐，在____年解密后适用本授权书。

本论文属于

不保密☐.

（请在以上方框内打“√”）

作者签名：

指导教师签名：

日期： 年 月 日

日期： 年 月 日

Control of Virtual and Real Manufacturing Systems via AR/VR techniques

摘 要

虚拟学习和智慧工厂是制造业领域近年来的热点话题。Covid-19病毒的爆发更加提醒了人们数字镜像的重要性。如果工程师们不能接触到真实的机器，怎样才能远程操控它们？学生们只能通过网络学习，又怎样通过虚拟技术学习机器的操作？我们的项目旨在使用虚拟现实（VR）和增强现实（AR）技术来实时展示一个数控雕刻机的状态并模拟工件的雕刻过程，建立数控雕刻机的数字镜像。

通过我们的系统，用户可以使用VR手柄或手套来远程控制数控雕刻机，这些控制指令将实时传输并与VR端同步。同时用户可以从VR屏幕中观看到雕刻机的三维模型和工件的雕刻过程。不仅如此，用户也可以选择手动控制机器，所有的数据都会被程序记录下来并且传输到另一端，使得另一位用户能够实时观察到机器的状态。在VR系统的基础上，我们还设计了可以部署在移动设备上的AR应用程序。我们的程序会追踪特定的图形标签，随后将工件的三维模型显示在移动设备的相机图像里。实际应用时，用户可以不安装实际工件就可以通过手机程序在机器上观看工件雕刻的效果，达到虚拟操作的效果。

在这个项目中我们实现了一整个数字系统，涉及实时数据传输，三维建模和模拟，远程数据库维护，机器数据读取和交互，多平台应用程序开发等诸多技术。我们的项目证明VR和AR技术在虚拟学习和智慧工厂中巨大的应用潜力。在未来可以进一步被运用于更广泛的工业场景，助力下一代制造工业。

关键词：AR/VR, Unity, 制造机器, 数字控制, 数字镜像, 远程控制

Control of Virtual and Real Manufacturing Systems via AR/VR techniques

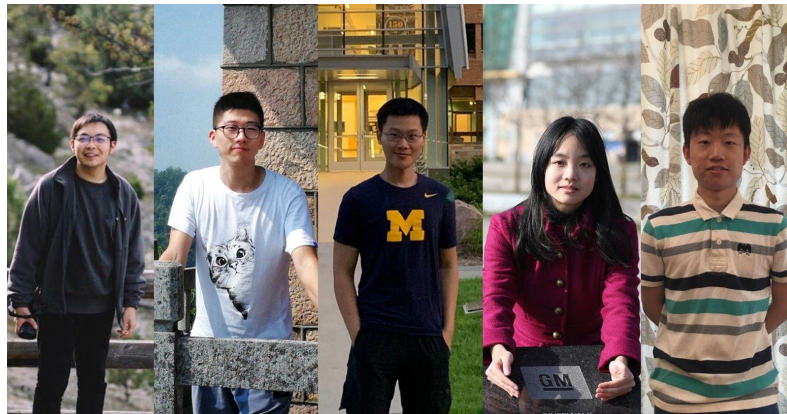
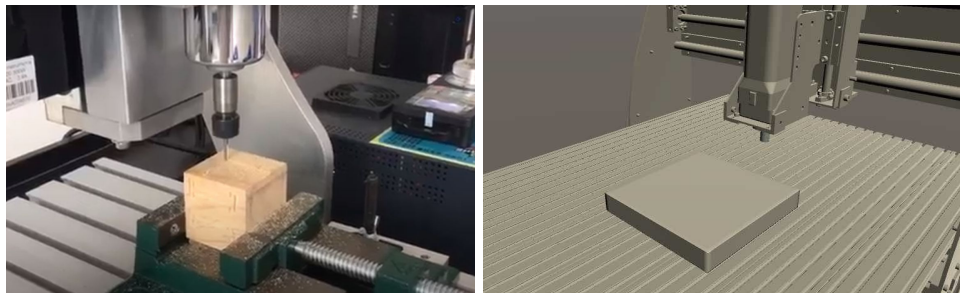
Abstract

Virtual Learning and Smart Factories have been heated topics in Manufacturing Industry. The importance of creating Digital Twins for manufacturing machines is becoming more evident after the breakout of the pandemic of covid-19. How could engineers manipulate machines from distance? How could students learn using manufacturing systems online? Our motivation is to use Virtuality Reality and Augmented Reality technology to Synchronously show and simulate the manufacturing process of milling artifacts.

With our VR system, users could use a VR handle or globes to control a CNC milling machine. Then the commands will be sent to the milling machine in real time and a 3D model of the machine as well as the artifact would show up in the VR headsets. Users could also see the milling process via our simulation. Additionally, the user could manually control the CNC milling machine while the data would be recorded and sent to other users, where they could watch the manufacturing process simultaneously. On top of the VR system, we implemented an AR application which could be deployed on mobile devices like IOS and Android smartphones. With this app, for instance, users could use their phone to watch the manufacturing process without manipulating a real machine. The app would track and locate specific tags and display the Virtual model on top of camera images.

Keywords: *AR/VR, Unity, Manufacturing Machine, CNC, Digital Twin, Remote Control*

Control of Virtual and Real Manufacturing Systems via AR/VR techniques



Course Name: VE/VM450, 20 Summer

Group Members: Zhiqi Chen, Tianyi Ge, Yue Wu, Yue Zhang, Junwei Deng

Sponsor: Prof. Mian Li

Institute: UM-SJTU Joint Institute

Date: July 19, 2020

Executive Summary

This project aims at building Digital Twins for a CNC milling machine using VR/AR technologies, which could boost the development of virtual learning and smart factories. Specifically, we built a digital system that consists of a virtual model representing the real world objects, a software that maintains the database and communication between the model and machine, a VR/AR environment where users could interact with the machine. By comparing benchmarks and studying the customer requirements and engineering specifications, we set up goals for key indices like refresh frequency, average latency, and financial cost, etc. We then divide our projects into three stages, including researching concepts, design and implementation of VR/AR input control system and CNC real-time feedback system, and iterative improvements of the system based on tests.

We made careful decisions for each sub-functionality and strictly executed the manufacturing plan to limit our budget, including socket connection, database deployment, CNC milling machine installation, 3D Unity model for simulating both the milling machine and the workpiece, all of which, as a remote control system, provides our users with an immersive experience. Augmented reality function, built on Vuforia, is capable of simulating the manufacturing process of a workpiece to facilitate production efficiency. For the test results, most of our quantitative requirements are fulfilled. The latency level and update frequency is within acceptable limits.

In summary, we developed our project using selected concepts and strictly follow the engineering requirements, within the budget expectation. Our final delivery has fulfilled the expected functionality requested by our sponsor, but we can still foresee sustainable improvements to be developed from our system.

Keywords: AR/VR, Unity, Manufacturing Machine, CNC, Digital Twin, Remote Control

Contents

Executive Summary	2
Contents	3
Chapter 1 Introduction	5
1.1 Background and problem description	5
1.2 Concepts behind the project	6
Chapter 2 Specifications	7
2.1 Customer Requirements	7
2.2 QFD Analysis	9
2.2.1 Research on related projects	10
Chapter 3 Concept Generation	12
3.1 Brainstorming and Concept Diagram	12
3.2 Concept sub-solutions	13
Chapter 4 Concept Selection	15
4.1 Weight Factors	15
4.2 VR headset	16
4.3 3D Software	16
4.4 Transmission	16
4.5 Storage	17
4.6 Control software	17
4.7 CNC Machine	18
Chapter 5 Final Design	19
5.1 Chosen concept	19
5.2 Engineering Design Analysis	20
5.2.1 AR Mechanism	20
5.2.2 Artifact Simulation lattice	20
5.3 Design Description	21
5.3.1 Virtual Reality part	21
5.3.2 Augmented Reality part	23
5.3.3 Socket library compilation	24
5.3.4 Database connection and authentication	25
5.3.4.1 Server End	26
5.3.4.2 Client Ends	27
Chapter 6 Manufacturing Plan	28
6.1 Virtual Reality part	28
6.1.1 HTC Vive Pro	28
6.1.2 SteamVR	28
6.1.3 Unity	29
6.2 Augmented Reality part	29
6.3 Socket library compilation	32
6.4 Database connection and authentication	32
6.5 CNC Machine setup	33

Chapter 7 Test Results	36
7.1 Validate update frequency and latency	36
7.2 Validate deviation rate	38
7.3 Variables that are not validated	38
Chapter 8 Engineering Changes Notice	39
Chapter 9 Discussions	40
9.1 Strengths and weaknesses	40
9.1.1 Strengths	40
9.1.2 Weaknesses	40
9.2 Potential improvements	40
9.2.1 Computer hardware resource requirements	40
9.2.2 The impact of camera resolution on Augmented Reality function	41
Chapter 10 Recommendations	42
Chapter 11 Conclusions	43
Chapter 12 Acknowledgements	44
References	45
Chapter 13 Appendices	46
13.1 Socket library server-side code in C	46
13.2 Mach3 control and communication code in VB	48
13.3 Partial Unity 3D Model Shader code in C#	50
13.3.1 CylinderCut.cs	50
13.3.2 VertexGenerator.cs	50
13.4 Partial Database connection code in PHP	53
13.5 CNC Milling Machine Parameters	54
Bill of Materials	55
Bios	56
Yue Wu	56
Zhiqi Chen	56
Junwei Deng	56
Yue Zhang	56
Tianyi Ge	57

Chapter 1 Introduction

1.1 Background and problem description

Nowadays, smart manufacturing becomes more and more critical in the mechanical engineering field, and the “Digital Twin” concept is one of the popular implementation methods. Digital twin concept consists of a real-world industrial good and a virtual model simulated by computer software. The virtual model is craftly designed and automated to simulate the status and send user controls to the real-world industrial goods. Moreover, AR and VR techniques have been developed greatly in recent years. 3D virtual models can be constructed and seen in AR/VR glasses.

Under this pandemic of covid-19, for instance, some engineers have no access to manufacturing machines. Or, imagine new students want to learn to use and grasp some machines in the factory, how can we learn this online? So our motivation is to use virtual reality technology to synchronously show the manufacturing process and simulate the processing of a workpiece. Another advantage of AR simulation is to validate a milling program without wasting materials, like wood or metal.

In our specific case, we will focus on the design and implementation of the digital twins of a CNC milling machine with AR and VR techniques. CNC milling machine is a complex engineering machine which is hard to monitor and dangerous to directly operate for students and even experts. The project is aimed for these design problems.

1. CNC milling machine is dedicated and has good support for numerical control, which is suited for an automated factory with lower level of manual intervention. Such factories would benefit a lot financially from remote supervision.
2. CNC milling machines are of high risk of accidents so that remote manufacturing reduces the horrible impact of factory emergencies.
3. New engineers or student learners might perform irregular operations, due to a lack of experience, or run an incorrectly compiled GCodeprogram, which would waste workpiece materials or even threaten the personal safety of themselves.

In this project, the CNC milling machine’s movement and status are monitored and simulated by the 3D virtual model and shown on the AR/VR glasses. Users can also use gloves and VR handles to control the CNC milling machine, including its open/close and operation parameters. The expected outcome can be listed as following:

1. A Unity model for CNC milling machine and its appendages which can be shown in AR/VR glasses.
2. A communication system between the CNC milling machine and the virtual model with mach3 scripts, web database and other scripts.

3. A VR interactive code and environment which makes users interact with the model for CNC milling machine with handles.
4. An AR interactive code and environment which makes users can see the simulation and demo of material on the CNC milling machine under a series operation.

1.2 Concepts behind the project

The concept behind our project is called “Digital Twins”. “Digital Twins” is a pair of objects while one of them is realistic and in the physical world, and the other is a virtual software model which shows the status of the physical object in a real-time manner. It consists of three parts: a real space object, a virtual space object and the information conveyed between them. Our design will focus on the virtual reality modeling synchronization, implementation of highly efficient information communication protocols, and the precise rendering on the augmented frames.

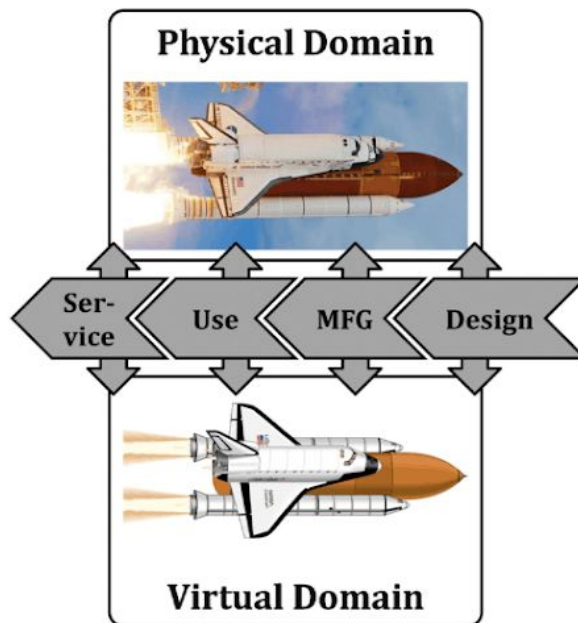


Fig. 1-1a Concept of “Digital Twin”

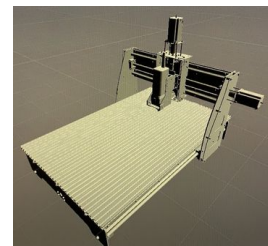


Fig. 1-1b “Digital Twin” relation in our project

Chapter 2 Specifications

2.1 Customer Requirements

Based on industrial surveys and VR/AR annual reports [1, 2, 3, 4], we finalize our customer requirements, including

1. Real-time synchronization: as the key spirit of “Digital Twin”, real-time synchronization with the real manufacturing machine fosters remote monitoring and operations.
2. Precise positioning (AR): when rendering an augmented object onto the surface of a milling machine, precision is the most consequential feature that users care about. A precisely placed object (workpiece) is not only a manifest reflection of the manufacturing process but also a rehearsal to pre-check the soundness of Gcode commands, thus preventing potential hazard.
3. Immersive experience: Unlike a 2-dimensional screen, virtual reality provides immersive experience and high interactivity. According to the survey, users incline to expect more degree of freedom to interact with the 3D models.
4. High software portability: since we are developing an entire VR/AR system entailing different components, it’s favorable to obtain a flexible abstraction design to be available on multiple devices.
5. Low financial cost. It’s a common requirement of customers among every industrial product. Since the goal of this design is to facilitate numerical manufacturing with virtual reality devices, the extra costs are almost negligible. The financial cost is expected to be kept as a low level, especially if the users could provide their own machines and devices, which covers a great share of expenditure.
6. Low learning cost: the interface and operating procedure should be as intuitive as possible. The ease to operate on the machine, trivial as it may appear, is enlarged for engineers who, for example, met emergencies. In that case intuitive operations would save money and life. Otherwise, it conflicts with our motivation to provide users with immersive experience and in-time “brake” control when accidents happened.

Analyzing and decomposing the customer requirements into engineering languages help us understand fine-grained subfunction components, thus divide-and-conquer the behemoth system. We first brainstormed a pool of engineering specifications, either quantitative or qualitative. It’s because qualitative specifications can often be further broken apart to detailed quantitative components. Next we map the requirements to the specifications, following a many to many relation. If during this process new specifications occurred, we add it to the specification pool and check if any other requirements would correspond to it.

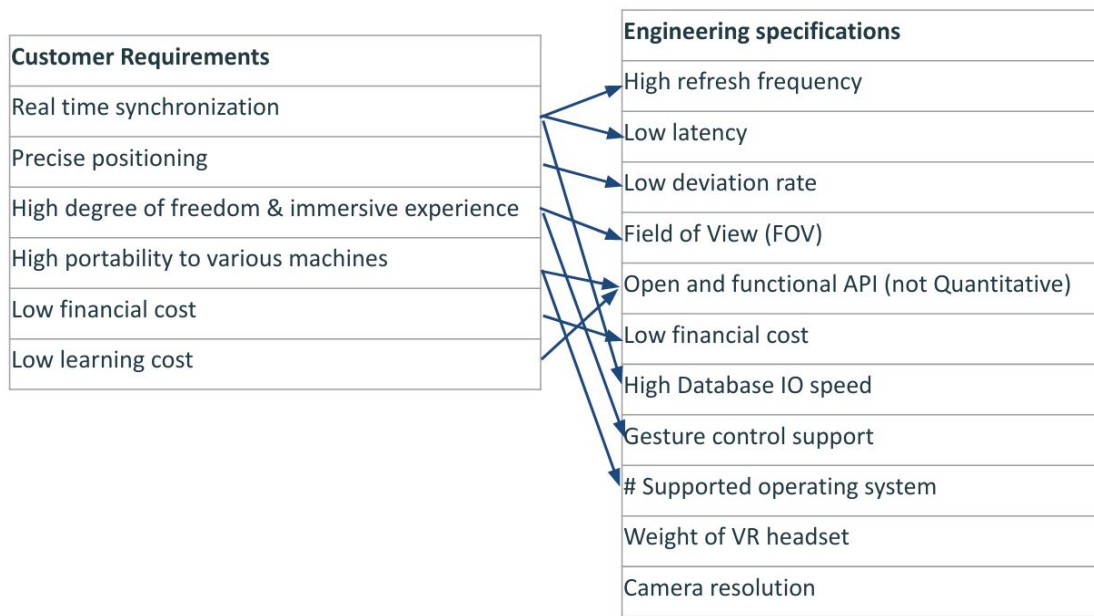


Fig. 2-1 Generation Process of Engineering specifications from Customer Requirements

Eventually we exclude two inconsequential engineering aspects (weight of VR headset and camera resolution) and the only non-quantitative specification. The finalized engineering specifications include the following quantitative variables:

Engineering Specifications	Measuring Unit	Target values
Refresh frequency	Hz	≥ 20
Avg. latency	ms	≤ 200
Object deviation rate	%	≤ 7
Field of View	degree	≥ 200
Financial cost	RMB	$\leq 7,000$
Database speed	ms	≤ 30
Handle control support	bool	1
# Supported operating systems	#	≥ 2

Table. 2-1 Quantitative Engineering Specifications and our target values

Notice that the handle control support is actually a bool variable, which means either true or false. We treat it as an engineering specification as well, but it is too trivial to contribute enough to our design selection and validation. The expected numbers are referenced from surveys and competitor

products as mentioned above [3, 6]. For example, the two variables, update frequency and latency, are positively correlated, which means higher frequency induces higher latency. The baseline of 20Hz, which is a little below the capability of Siemens digital twin product [6], however, is decided by considering our geo-distributed scenario and latency requirements.

2.2 QFD Analysis

After determining customer requirements and engineering specifications, the next step of our design is to evaluate the importances of Customer Requirements, and the correlation between Requirements and Specifications, thus facilitating us to make appropriate decisions to trade-off.

The customer requirements are compared pairwise, yielding the binary importance, as listed in the QFD chart. To facilitate understanding, we sort them in the descending order of importance level.

Real-time sync	1	1	1	1	1											5	33%
Precise positioning	0					1	1	1	1							4	27%
Immersive experience		0				0				1	1	1				3	20%
High software portability			0				0			0			1	1		2	13
Low financial cost				0			0			0		0		1	1		7%
Low learning cost					0			0			0		0	0	0		0%
Total																15	100%

Table. 2-2 Binary comparison of each customer requirements to determine the weight factor

Then, we clearly see a trade-off between refresh frequency and precision/latency. This is because when the virtual model requires more data per period of time, the computation load of control software is inevitably increasing, thus extending the processing time and latency to respond. Another example, to clarify, is that the Field of View is negatively correlated to financial cost because it's a hardware selection incurring more expenditure on Virtual Reality equipment. Also, purchasing handle control equipment would increase the financial cost, while it's definitely worthy.

The importance of Engineer Specifications are the weighted sum of each specification, as listed "Absolute importance". The ranking results manifest that the variables associated with speed and precision are the leading factors, which will greatly affect the user experience. The significance of refresh rate and latency is self-explanatory because the entire real-time system would not be competitive, for example, if the VR displaying shows the machine status tens of seconds ago, or even one frame per second. Generally speaking, as a user-oriented design, mitigating time issues is our first priority.

Then, the competitive analysis we performed was based on the following benchmarking competitors: Siemens “Digital Twins” project, and the educational tool developed by Fab Lab.

2.2.1 Research on related projects

We have found two similar projects which can be our benchmark. The first project is the SIEMENS Plant Simulation project (left). It created a virtual environment in Unity to simulate a real CNC manufacturing process. The communication between the physical machine and the virtual model is through both local and online databases. Although the project successfully presented the visualization part, it has low data transmission speed due to its communication structure and the virtual model has low resolution. Also, the project is a one-way simulation which means it doesn't have a remote control function. The second project is the Fab Lab Connection (right). It is a digital design tool and an educational platform on which people can simulate and visualize the working of CNC machines. Since it's only a virtual platform, it doesn't have physical lab support or real-time simulation. Here summarize the shortcomings of both projects:

1. Low data communication speed (high latency).
2. Low model resolution.
3. No remote control access.

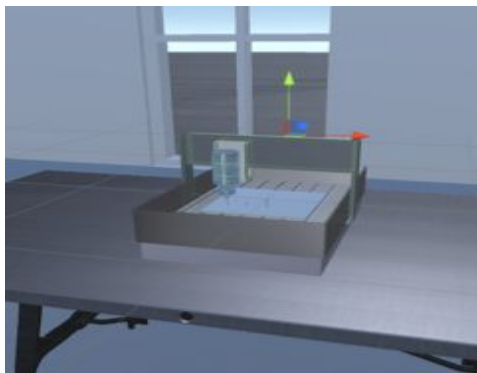


Fig. 2-2a SIEMENS VR product demo



Fig. 2-2b Fab Lab VR product demo

Therefore, there are lots of points we can improve in our project:

1. Optimize connection structure and backend server to reduce latency.
2. Utilize realistic Solidworks models to improve resolution.
3. Apply handle control to allow remote access.
4. Add AR auxiliary to make CNC operation easier.

The comparison results show that Siemens outperformed the Fab Lab in many aspects like real-time synchronization, high software portability, and precise positioning, but its interactivity can be further improved by adding support for control operations like handle control.



Fig. 2-3 QFD analysis table of our specifications

Chapter 3 Concept Generation

3.1 Brainstorming and Concept Diagram

The process of concept generation begins with group brainstorming. We set a strict time limit and brainstormed individually. After the individual brainstorming we shared the thoughts and decided the concept diagram sketch.

The sketch diagram shows that the concepts are generally divided into three categories. The green circles are user-side concepts; the blue part is networking concepts; and the red part is the CNC machine concepts. They are inter-connected with both control and data flow.

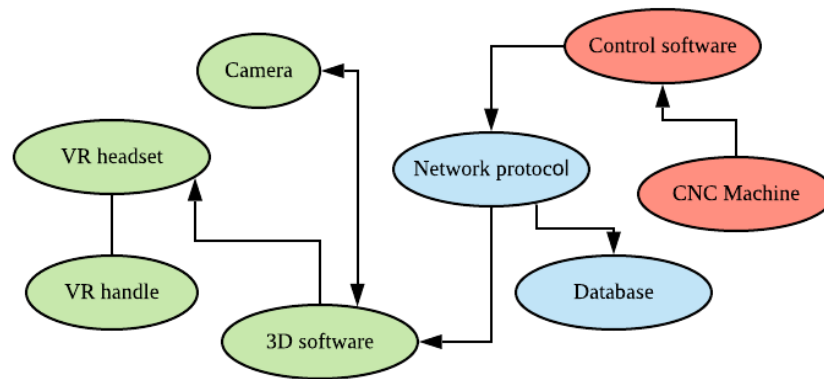


Fig. 3-1 Sketch diagram of concepts after brainstorming

After integrating and polishing the sketch above, we had the final concept diagram with specified signal flows. As shown below, the users are decoupled to two sub-categories: remote users and local users. The remote users are able to view the 3D model status through VR headset, and give handle control signals (red arrows) to the real machine, such as manually moving the drill along three axes, loading Gcode file, and emergency stop, etc. On the other hand, the local user who is physically next to the machine can view how the real-time machine model is moving and cutting the workpiece material. The specific choice of AR platform, which will either be integrated with VR or separated, will be elaborated in the next section.

Transmission part is responsible for data transmission and persistence. All the packets/signals will be recorded and stored in a cloud database. It seems like a basic concept but is also a complicated one because there's a diversity of network protocols. Therefore, an appropriate protocol choice is highly critical to boost the system performance.

At last, the CNC machine and control software, as a whole, compose the endpoint which accepts all the incoming commands and sends feedback to the client side. Another role of a CNC machine is that it provides camera video to the AR platform so that a virtual workpiece will be presented to the AR user with high-quality rendering. Integrating those three major parts, the entire closed-loop system can reliably provide control and feedback service via VR/AR.

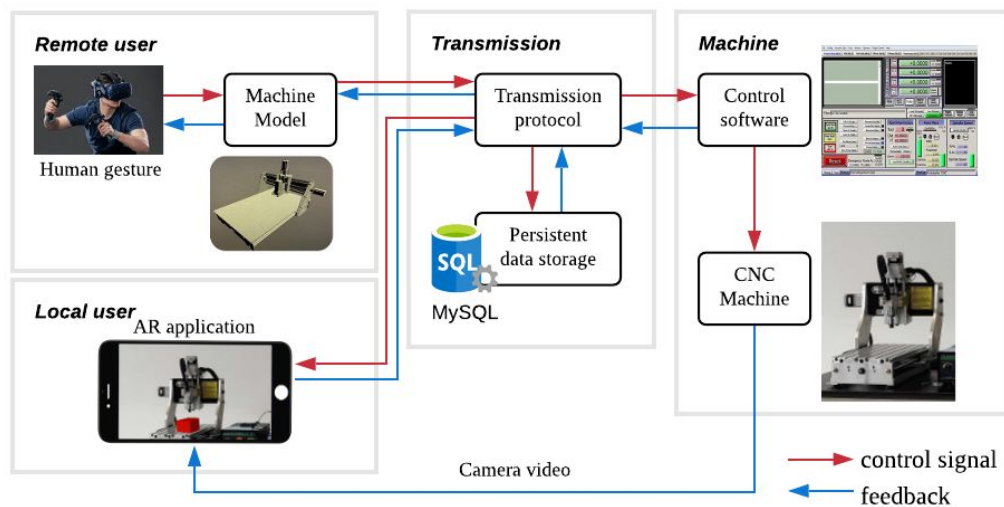


Fig. 3-2 Polished concept diagram. (red=control signal; blue=feedback)

3.2 Concept sub-solutions

After the morphological analysis and function decomposition, broader research is conducted to prepare adequate engineering choices for the concept selection process. Each concept has abundant products/tools on the market.

Here provides a brief introduction to the most important concept sub-solutions. For example, VR headset is a quite hot topic both in industry and among gamers. Here we consider a choice among HTC Vive Pro, Playstation, and Oculus Rift, as three popular products based on a survey in 2019[3]. For 3D softwares, Unity 3D and Solidworks are well-known for their compatibility and universal popularity. Transmission tool and storage, which directly determines our transmission performance, should be examined carefully. Common strategies for communication are TCP socket connection, Amazon IoT core, and file read/write functions provided by VisualBasic. The VisualBasic is taken into consideration because it's the only language that Mach3, the most widely-used control software, supports for programming. On the other hand, Amazon DynamoDB supports high concurrent database operations, while is not as common as MySQL or MongoDB.

With this carefully collected table, we are prepared to proceed and winnow the best concept combination for this very project. The detailed discussion about their respective pros and cons will be presented in the next section.

No	VR headset	3D software	Transmission	Storage	Control software	CNC
1	HTC Vive pro	Unity 3D	Socket conn.	MySQL	Mach3	Milling
2	Playstation	Solidworks	Sych. text file	DynamoDB	ArtCAM	Aishi Lathe
3	Oculus Rift	-	Amazon IoT	MongoDB	UG	Robot arm
4	-	-	-	Redis	Type3	-

Table. 3-1 Available choices of sub-functions before concept selection

Chapter 4 Concept Selection

Concept selection process involves the decision making on the concepts we finally used for this project. Since our project has many components, we have prepared 2 - 4 concepts for each component as shown in Table 4-1. With all these sub-solutions we can have 864 combinations for our project. To simplify the selection process, we will carry out the selection methods within each sub-solutions space and the relative reasons across different sub-solutions will be considered as well. The reasons will be based on both absolute criteria and relative criteria.

4.1 Weight Factors

Weighted Decision Matrix will be a critical and important relative criteria we will use as a selection method. First, we need to decide the weight factors for each design criterion. A hierarchical objective tree will be used for this part. Engineering specifications are assigned their own weight factors.

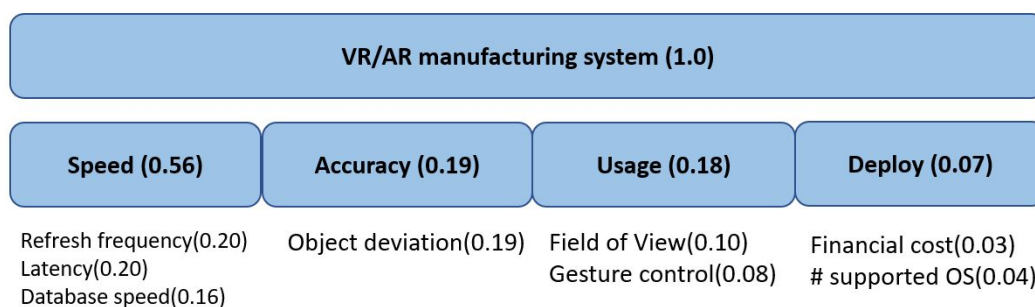


Fig. 4-1 Weight Factor Hierarchical Objective Tree

We first divide the project criteria to four parts. For the speed, the response time of the control command and the timely feedback should be considered. We defined three criteria for this section. Refresh frequency refers to the virtual model update times per second when the CNC machine is sending back real time status. Latency refers to the waiting time between commands sent to the action on a CNC machine. Database speed is measured as the read and write speed for the database we use. We will give further descriptions on it in later sections. For the Accuracy, we want to have a vivid model for users to interact with and supervise in the virtual environment. Object deviation refers to the difference of position and size between real object and virtual object. For the usage, user's experience and interaction in the VR/AR environment should be considered. Field of view refers to the horizon when users take on the VR glasses. Gesture control means that we need to make our

system have easy compatibility with gesture control. For the deploy, we need to make the system easy to deploy on every OS with low cost.

4.2 VR headset

As stated in Section 3.2, we have three choices among HTC Vive Pro, Playstation, and Oculus Rift for VR headset. HTC Vive Pro is a VR headset released in 2015 that was developed by HTC and Valve, which makes it support SteamVR natively. Playstation VR headset is developed by Sony and it is designed for the Playstation host. Oculus Rift was developed by Oculus VR and has a relatively close software environment. We choose HTC Vive Pro for two reasons. First is the feasibility reason. We have one in the lab and we don't need to purchase it again. Second is the reason for technology readiness. We can develop and simulate the VR interface directly through SteamVR on any laptop since it has the native support. More tutorial and problem solving experience can be found because of the broad and easy-access developing platform. Playstation VR headset is only designed for the Playstation host, which requires that the developers and the users should purchase a headset and a host. It is beyond our budget and too complex for the users. Oculus Rift has a close developing environment and worse display quality.

4.3 3D Software

As stated in Section 3.2, we have two choices among Unity 3D and solidworks. Solidworks is a professional software for computer aided design. Still it is not designed for VR/AR development and the model it makes still needs to be imported to Unity 3D for interactive interface design and implementation. With these disadvantages, Solidworks is excluded from our final choice. We will use Unity 3D for virtual environment implementation.

4.4 Transmission

As stated in Section 3.2, we have TCP socket connection, Amazon IoT core, and file read/write functions provided by Visual Basic. All of them meet with the feasibility and readiness requirement and can be implemented by us. We did some experiments and compared their performance on the engineering specifications by scoring matrices as shown in Table 4-1.

Design Criterion	Weight Factor	Unit	Sync File			Socket Connection			Amazon IoT		
			Value	Score	Rate	Value	Score	Rate	Value	Score	Rate
Refresh frequency	0.2	/s	10	8	1.6	100	10	2	75	9	1.8
Latency	0.2	ms	100	9	1.8	100	9	1.8	200	7	1.4
Object deviation	0.19	exp	Good	10	1.9	Good	10	1.9	Good	10	1.9
Field of View	0.1	exp	Good	10	1	Good	10	1	Good	10	1
Financial cost	0.03	CNY	0	10	0.3	0	10	0.3	1000	5	0.15
Database speed	0.16	ms	50	8	1.28	10	9	1.44	10	9	1.44
Gesture control support	0.08	bool	yes	10	0.8	yes	10	0.8	yes	10	0.8
# supported OS	0.04	exp	Excellent	10	0.4	Good	8	0.32	Good	8	0.32
					9.08			9.56			8.81

Table. 4-1 Weighted Decision Matrix for Transmission Sub-solution

In Table 4-1, we find that socket connection performs better in the speed relative criteria such as refresh frequency and latency and has a little shortcoming in compatibility. Sync file has a better performance on deployment criteria such as OS support. Speed is much more important in our weight factors. So, we choose to use socket connection.

4.5 Storage

As stated in Section 3.2, we have MySql, DynamoDB, MongoDB and redis. We involve the database as a transfer station for the position of the cutter and the orders sent from the users. The data amounts are small and structured. MySql is one of the most popular database systems with SQL support. It is an open source lightweight relational database, which makes it quite easy to fit our demands. DynamoDB is developed by AWS as a NoSQL database, which is hard to use. MongoDB is a NoSQL database as well. Redis is a Key-value database. Since MySql meets with our data so well we use it as our final decision.

4.6 Control software

As stated in Section 3.2, we have Mach3, ArtCAM, UG, Type3. For control software, we refer to the software that controls and supervises the CNC machine. The choice is highly related to the choice of the CNC machine we will carry out our project. So, we only need to make sure the software can be programmed to extract the real time CNC machine status and control the CNC machine through commands like gcode. We choose Mach3 as our final decision. Still, Mach3 is an old software which prefers old OS like Windows XP. This can be its shortcoming.

4.7 CNC Machine

As stated in Section 4.2, we have Milling, Aishi Lathe and Robot arm. The choice is made by our sponsor since the milling machine is a typical CNC machine and the process to control a CNC milling machine can be easily transported to other machines.

Chapter 5 Final Design

5.1 Chosen concept

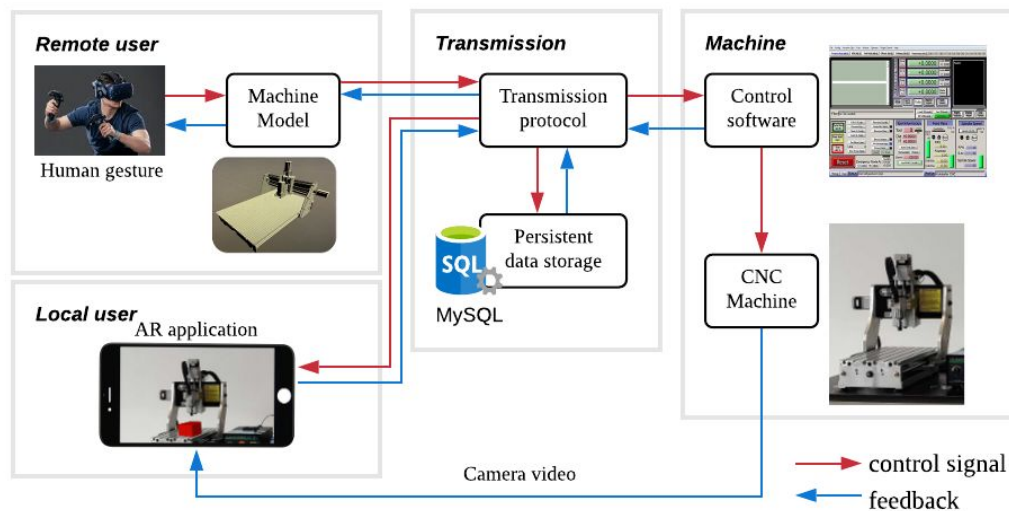


Fig. 5-1 Concept diagram of sub-function components

There are three main parts: user, transmission and machine. The remote users could control the machine by gesture and observe the machine with VR headset. The local user can use mobile APP to see a virtual workpiece rendered on the screen. The red arrow represents control signals input by the user; while blue arrows are real-time feedback from the machine. In order to synchronize the two ends, we need to build a robust transmission mechanism.

For the user side, we choose HTC Vive pro as the headset for VR since it is a commonly used VR headset with friendly handles and good user experience. We choose smartphones to be the AR terminal because that everyone can download an APP and access to the system. Unity 3D is chosen to build the model since it can make and simulate vivid 3D models.

For the transmission side, we choose socket connection to reduce the latency and raise the update frequency. Mysql is a stable and fast database that can transmit and store structured data in a short time.

For the machine side, we choose Mach3 to control the CNC milling machine, the software can read the real time milling machine status in a low latency.

5.2 Engineering Design Analysis

This project is a very comprehensive project which includes many scientific fields. For the user sides, we need to develop a VR user interface within the VR headset and a AR interface within the smartphone APP. UI/UX knowledge and ability to learn a new toolkit are critical here. We deploy VRTK and Vuforia toolkit here for VR and AR interface respectively. For the transmission sides, computer network knowledge and database knowledge are critical. For computer knowledge, we use socket connection as the communication protocol. For database knowledge, we host a web database for order and machine status storage. For the machine side, mechanical engineering and engineer's thought is important. Milling machines and Mach3 are typical ME tools and they may have strange and critical failures sometimes. We need to learn from the experience and fix the hardware bug as soon as possible.

5.2.1 AR Mechanism

We have several important engineering decisions on each component. The first is the AR development Mechanism. Vuforia is a popular commercial AR SDK built in Unity 3D. There are many functions supported by Vuforia and we need to choose one of them as our AR Mechanism.

For our project, AR users are mainly available to see the simulated artifact being processed by the milling machine in a virtual environment. The artifact should be located in the exact place that can interact with the cutter. We use the image targets function by recognizing a tag put on a plane to locate the exact location.

5.2.2 Artifact Simulation lattice

Artifact simulation is a computation-intense operation. To make an artifact has calculability, it is modeled as a lattice with many sampling points in the artifact and on the surface. The number of sampling points will affect the computational complexity directly. The points are sampled from the artifact iteratively. The number of points can be written as:

$$\text{Recurrence relation: } F(n+1) = 4F(n) - 6$$

$$\text{Initial state: } F(1) = 8$$

Eq. 5-1 Recurrence relation of the point sampling

Where F is the number of sampling points, n is the iteration number. For the first iteration, we initialize only 8 nodes. But according to *Master's Theorem*, the number of points is too large (6146) when it reaches a reasonable 6 iterations. if we don't make any change even for a laptop with an

Nvidia GTX 2070 GPU. We decided to use a sampling strategy only on the top surface. The formula will become:

$$\text{Recurrence relation: } F(n+1) = (2F(n)^{0.5} - 1)^2$$

$$\text{Initial state: } F(1) = 4$$

Eq. 5-2 Revised point sampling

The number of points is successfully reduced to 1089 on the 6 iterations. With this setting, the capacity of our computational power is adequate to run this task. If better computational resources are available, then of course we could deploy the first strategy so that the simulation results of the virtual workpiece would be much closer to the real manufactured product.

5.3 Design Description

5.3.1 Virtual Reality part

Based on our customer requirements, a user may observe virtual simulation synchronized with the CNC machine as well as sending commands to control the digital CNC machine twin in the VR environment. The design focused on the real-time feature to provide synchronization in all dimensions for monitoring.



Fig. 5-2 Our group member Junwei is testing VR functionality

As controlling input, we expected to control the milling machine in all three dimensions manually to feedback a series of positions of milling cutter. In this figure we display the three dimensions specified for cutting.

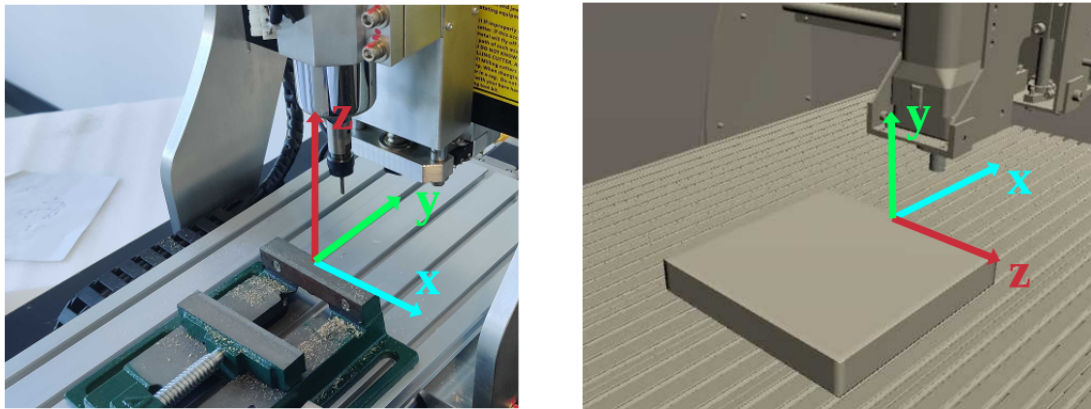


Fig. 5-3 The X/Y/Z axes of the cutting drill of real/3D model CNC Milling machine. Notice that the XYZ-coordinates are not identical thus requiring coordinate transformation

Also, the controlling panel in the VR environment is required to load a prewritten GCode program stored in the computer. The virtual milling machine should be able to display the simulation process and result, while pause or terminate the whole program anytime during the process.

Here, we design a control panel with four buttons and real-time position to realize the function illustrated above. To use this panel, we compiled a laser shot from the HTC Vive handler to shoot at and press down the buttons on the panel.

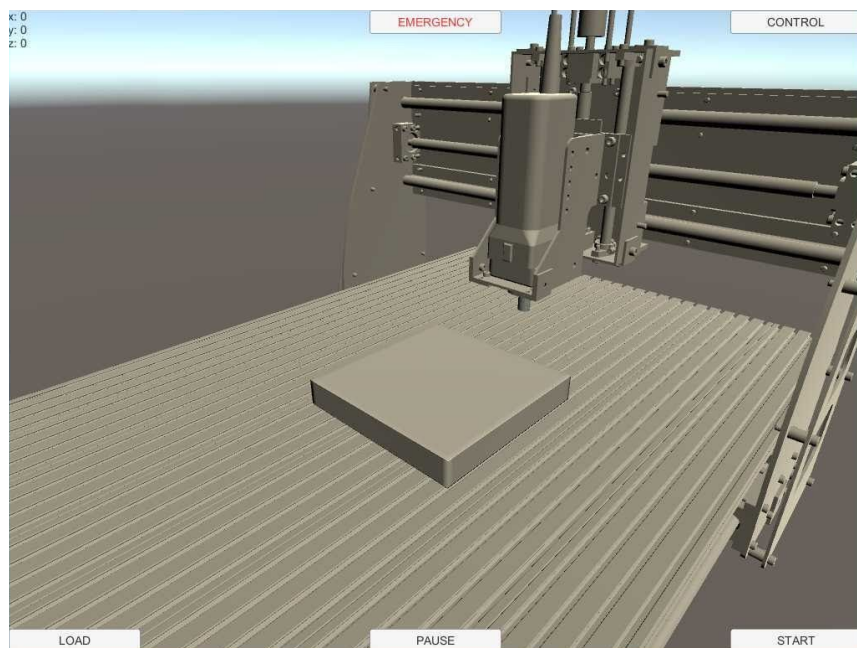


Fig. 5-4 The Virtual Reality user interface in Unity 3D, consisting of buttons CONTROL, LOAD, PAUSE, START, and EMERGENCY

1. Control button. After the control button is pressed, the user may manually manipulate the position of the milling cutter in three dimensions specified in figure above. They may send the current position of cutter whenever they want. Once the position is sent to the database, the real CNC machine will receive this position simultaneously and move its cutter.
2. Load button. After the load button is pressed, the computer will load a specific GCode program. While sending commands to activate the real milling machine, this computer also sends back multiple point positions continuously to provide simulation of the program.
3. Pause button. Pause button may suspend the working progress. When pressed again, the progress will continue.
4. Stop button. Stop button may terminate the working progress.

To fulfill the requirement of real-time simulation, we design a new algorithm to simplify cutting effects. Below we illustrate the logic of this algorithm in Unity. In Unity, the program would first generate enough mesh vertices on the surface of the piece. The generated vertices are displayed on the right of the figure below. Then, whenever the cutter contacts the vertices, the vertices will modify its height to display the cutting effect. This algorithm is simple enough to provide both real-time connection and authentic simulation.

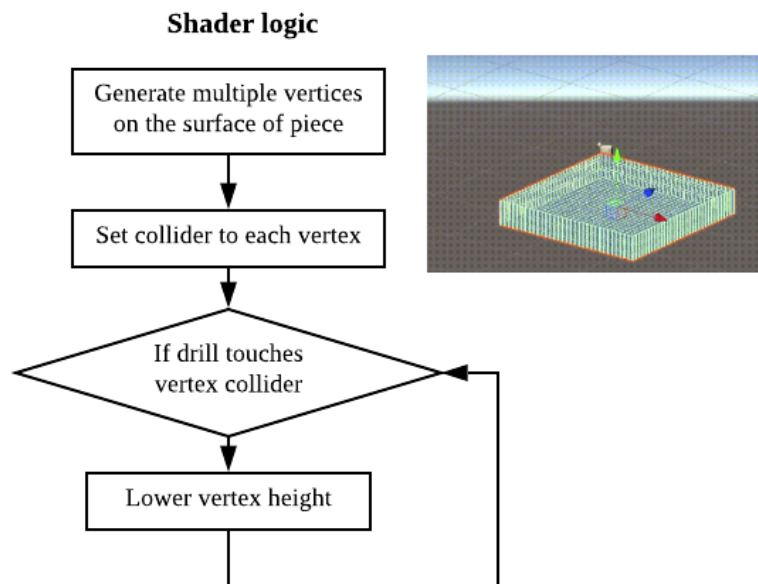


Fig. 5-5 The flowchart of Shader algorithm, and the demo effect of model vertices

5.3.2 Augmented Reality part

For the augmented reality part, we migrate the model and cutting control from the virtual reality part and feed them into an augmented reality environment. We choose to use Vuforia Augmented

Reality since it's one of the most popular AR environments in the Unity Engine and both desktop and mobile platforms are compatible with it.

To realize AR functions, we first need to set up a tag for detection and localization. We have designed a tag which is a combination of SJTU logo and a geometry pattern image so that it has enough character points for Vuforia to detect both the object and the direction. We use Vuforia online platform to generate a tag package and feed it to the Unity Engine to build an image target under which we give the model to visualize.

5.3.3 Socket library compilation

Socket programming is an intuitive way to connect two nodes on a network by assigning the role of server and client respectively to two ends. The server node listens on a particular port at an IP address publicly exposed, while the other socket (client) connects to the server. Once the connection is established, sending and receiving data, for example, a simple "hello" as the payload, are enabled between the two nodes. Socket programming provides both TCP and UDP protocol. Here we selected TCP protocol because a reliable data transmission is required here, while UDP does not guarantee the success of packet transmission. The detailed flowchart of connection establishing is shown below left.

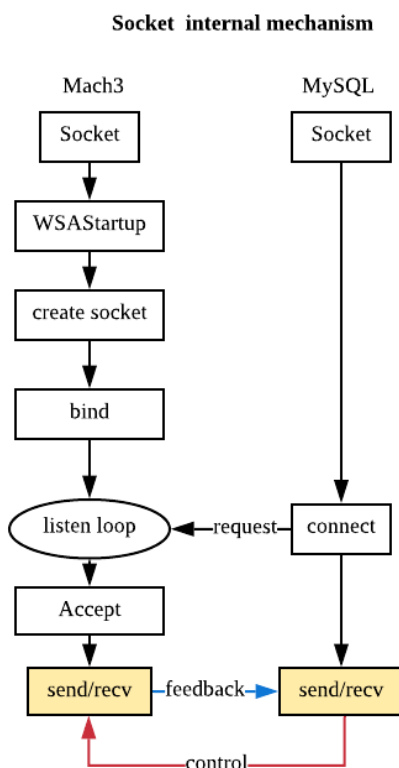


Fig. 5-6a Flowchart of socket connection establishment

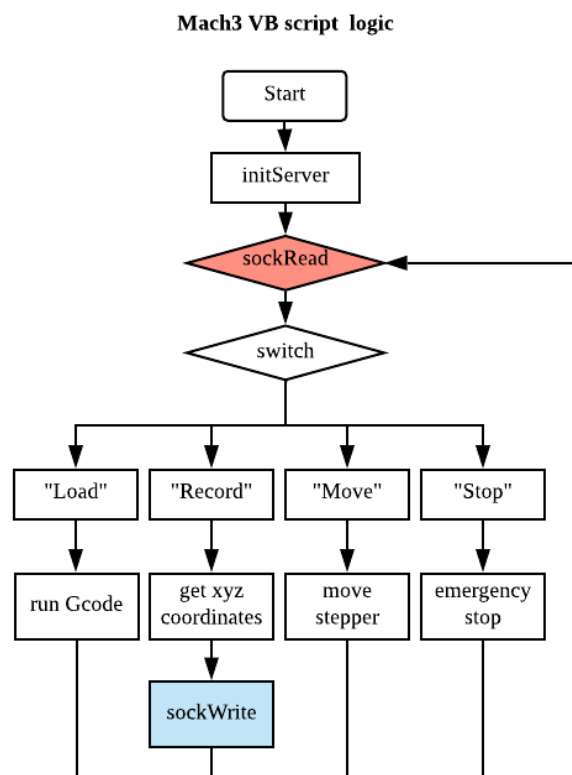


Fig. 5-6b Flowchart of Mach3 VB script

Now given the scenario of Mach3 and MySQL, it makes no significant difference who acts as the server or client. Thus, the Mach3 is chosen to be the server, waiting for requests, such as “get x coordinates”, from the MySQL node.

To enable the socket programming in VisualBasic, since it does not provide such socket APIs, an elegant solution is to compile a socket library written in C by ourselves. VisualBasic script of Mach3 is expected to call the following APIs:

1. “initServer” to start listening on the specified port
2. “sockRead” to receive data from client (MySQL commands)
3. “sockWrite” to send data to MySQL, such as X/Y/Z coordinates of the drill

Also, now that Mach3 only supports Windows XP/Windows 7 32-bit, the libsocket.dll should use Win32 system SDK instead of POSIX API on Linux. For example, the library we built should include “winsock3.h” instead of “sys/socket.h”, otherwise it will not be compatible with Mach3 on Windows. With the socket library available, the communication between Mach3 and database is ready. The flowchart above right shows the logic of Mach3 working principle, which supports four types of commands:

1. “Load” starts the predefined Gcode program and runs on the milling machine
2. “Record” gets the X/Y/Z coordinates from Mach3 and sends back to the database
3. “Move” enables manual control of the drill
4. “Stop” triggers emergency stop, in case accidents happened

There are more supported commands like “Pause”, “Rewind”, etc. They are all clearly stated in Appendix 13.2.

To reproduce this prototype, one needs to put the compiled socket library in the same directory with VB script. Once the VB program is started, MySQL database can safely connect to the Mach3 node. At this point, the end-to-end connection is complete.

The socket API of windows is free to use, thus incurring no financial cost.

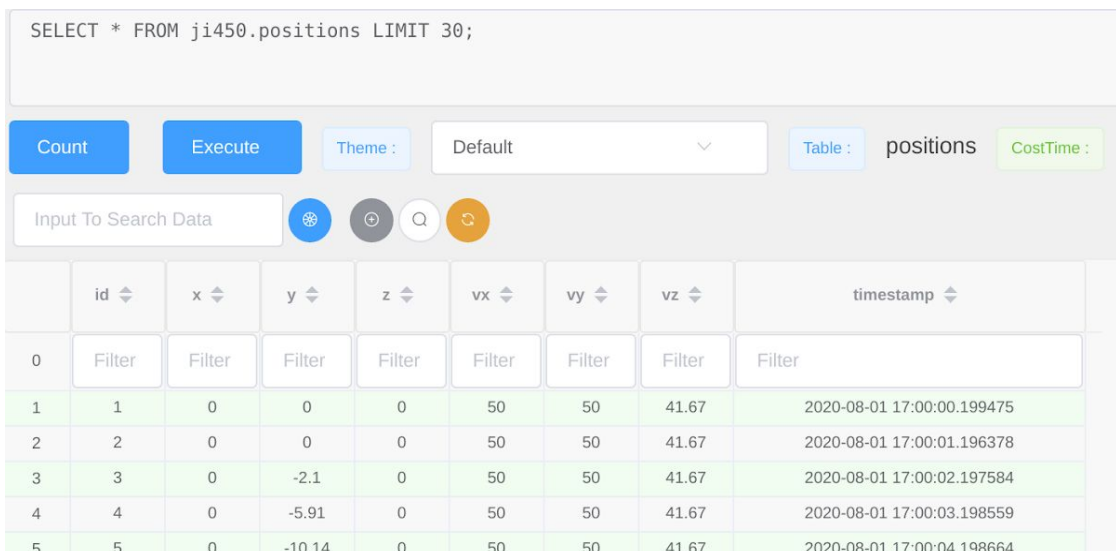
5.3.4 Database connection and authentication

Due to stability and efficiency requirements, we choose HuaweiCloud Platform MySQL as the intermediary of our end-to-end connection. It gives us full access to database operations as well as

security authentication. Since we have both online servers and various local clients, we will use different connection and authentication methods.

5.3.4.1 Server End

We locate a virtual machine on HuaweiCloud Platform which is the server of MySQL database. It gives us a static IP address so that we can have a stable connection to the server anywhere in the world. However, we also want our server and data to be secure so that there won't be any insertion attack on our database which may cause damage to real world CNC machines. We have several options: SSL encryption, private IP authentication, public IP authentication and proxy authentication. Based on the Unity framework on which we build our program, SSL encryption and proxy authentication are not compatible. And between private and public IP authentication, since our CNC end client is fixed in the lab which has a static public IP address and we want our users to connect to the database under any permitted web environment, we finally decide to use public IP authentication. We can set a white list IP address (for example, education webs) on the platform to limit user access.



	id	x	y	z	vx	vy	vz	timestamp
0	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	0	0	0	50	50	41.67	2020-08-01 17:00:00.199475
2	2	0	0	0	50	50	41.67	2020-08-01 17:00:01.196378
3	3	0	-2.1	0	50	50	41.67	2020-08-01 17:00:02.197584
4	4	0	-5.91	0	50	50	41.67	2020-08-01 17:00:03.198559
5	5	0	-10.14	0	50	50	41.67	2020-08-01 17:00:04.198664

Fig. 5-7a Sample MySQL records from the “JI” Gcode program

```

1 CREATE TABLE `positions` (
2   `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
3   `x` float NOT NULL,
4   `y` float NOT NULL,
5   `z` float NOT NULL,
6   `vx` float NOT NULL,
7   `vy` float NOT NULL,
8   `vz` float NOT NULL,
9   `timestamp` timestamp(6) NOT NULL DEFAULT CURRENT_TIMESTAMP(6) ON UPDATE CURRENT_TIMESTAMP(6),
10  PRIMARY KEY(`id`)
11 ) ENGINE=InnoDB DEFAULT CHARSET=utf8

```

Fig. 5-7b Creation of MySQL “positions” table

5.3.4.2 Client Ends

1. Desktop

On the desktop platform (MacOS, Windows and Linux), we choose to use MySQL connector .dll file to help us connect to the database. All three operating systems support Mono or .NET platforms so that all three are compatible to the connector. Due to public IP authentication, we can securely compile the connection IP address and the access info into our program to connect the database directly.

2. Mobile platform

Since both Android and iOS don't support default MySQL connector platform or direct database connection, we have to change the connection method to php connection on mobile devices. We utilize Huaweicloud Platform MyPhpDomain support and App Engine function to deploy a php website on the same virtual machine on the platform. With this static address website, we deploy another php query request api which will return a html page that contains the position info we need to simulate our model. Therefore, we can use the WebRequest package (WWW) in Unity to access the php api we develop and retrieve database data.

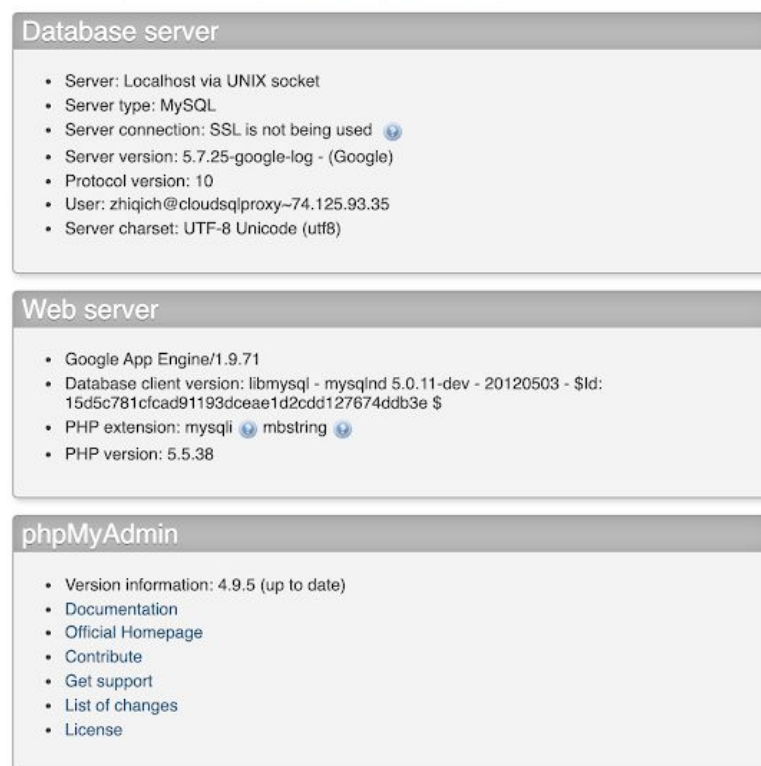


Fig. 5-8 Static web page for database connection on mobile device

Chapter 6 Manufacturing Plan

6.1 Virtual Reality part

Here we select HTC Vive Pro as VR device and use SteamVR in Unity to construct VR environment. This combination is technically mature for VR Input and real-time control, while there is one HTC Vive device available in the lab, which can be applied in plenty of VR projects. In consideration of both feasibility and compatibility, we apply this combination for environment construction. All material used is listed in following table:

Materials	Manufacturing process	Manufacturing Status
HTC Vive Pro (sponsored)	Install VR hardware device	Complete
SteamVR	Develop VR environment in Unity	Complete
Unity	Setup model and control panel, connect with database	Complete
Total budget	139 ¥	

Table. 6-1 Material setup for Virtual Reality equipments and softwares

6.1.1 HTC Vive Pro

We use the existing HTC Vive Pro device for development. To provide a stable video connection, we bought a mini-dp to mini-dp cable to connect to our own computer, which cost us 139 ¥. For installation, we set up this VR hardware device in the lab. After testing, this VR device is able to provide a stabilized monitoring environment.

6.1.2 SteamVR

SteamVR supports HTC Vive as its VR hardware system. To interact with the designed controlling panel, we developed a laser system with SteamVR to contact buttons. SteamVR successfully connects Unity with hardware devices to provide VR experience. In the budget aspect, the SteamVR plugin is free in the Unity Asset Store, therefore it doesn't cost.

6.1.3 Unity

As a mature real-time development platform in both VR and AR environments, we chose Unity to support real-time command and reaction for our user interface. We built up the model in Unity and created a controlling panel UI. Also we applied MySQL API to connect to an online database, which will be referred to in 6.4.3 and 6.4.4. Unity obtains a free version for student developers, so the budget for it is zero.

6.2 Augmented Reality part

Considering the portability and feasibility, we choose mobile devices like smartphones and Tablets, instead of the Vive Headsets in vr settings. Here is a list of materials used for our AR application.

Materials	Manufacturing process	Manufacturing Status
Mobile Device	Compile IOS/Android Apps	Complete
Vuforia (software)	Develop with Unity	Complete
Tags	Design Shapes	Complete
VR Components	Transplant from VR part	Complete
Total budget	0 ¥	

Table. 6-2 Material setup for Augmented Reality equipments and softwares

And their relationship is as shown in Fig. 6-8.

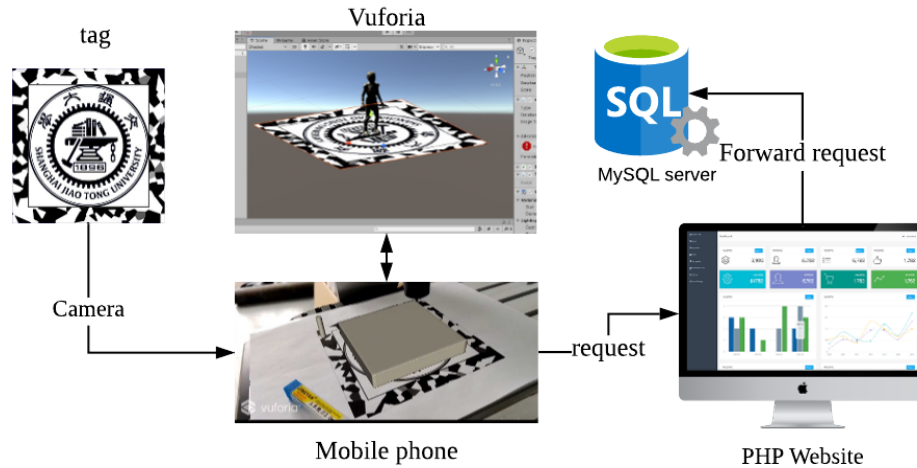


Fig. 6-1 AR Manufacturing diagram

We utilize the popular Augmented Reality environment, Vuforia, to manufacture our augmented reality applications. This application should be able to produce real time 3D modeling of a milling machine and display the artifact in a real world setting. Since vuforia has a supporting pack in Unity, we can easily migrate the models, shaders, and other components from VR to AR environments, which could help us satisfy most of the engineering needs.

1. Mobile Device

Any Smart phone or Tablet that uses IOS/Android System can be used as the hardware platform for our AR application. We have conducted real augmented reality experiments on both an Iphone X (IOS) and Xiaomi (Android).

2. Vuforia

We use [Vuforia Engine 9.2.8](#), which is an integrated free software toolkit that could develop and deploy AR applications [11]. The key feature it provides is algorithms that could detect and locate target images by camera inputs and an interface for developers to deploy 3D models on the desired location.

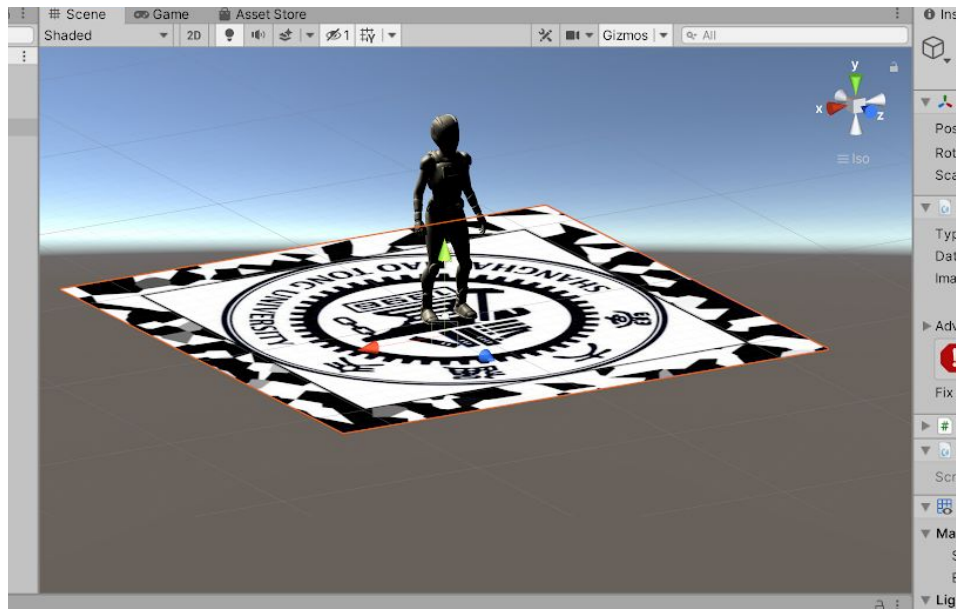


Fig. 6-2 Vuforia Develop Environment in Unity

3. Tags

Since Vuforia's algorithm has a low tolerance for the target image shape, we designed a specific tag that could well get recognized by the mobile device cameras. To achieve high detection rate and localization accuracy, we first generate rectangular voronoi diagrams as Fig. 6-10a.

Considering that voronoi with high density could lead to computation waste, we do tests and come up with a final design as

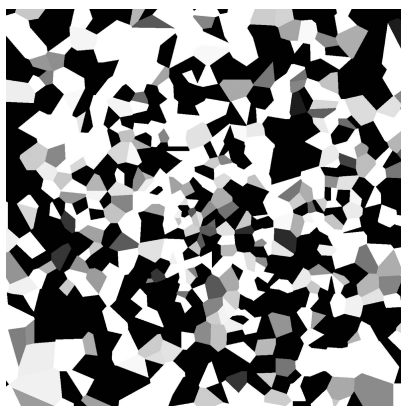


Fig. 6-3a Rectangular voronoi diagrams



Fig. 6-3b Tailored voronoi diagrams

These tags could be easily printed on paper and pasted on flat surfaces. With this tag, any CNC Milling machine is able to support such AR rendering, thus the high software portability is guaranteed.

4. VR components

Most of the parts including models, shaders, control scripts, except the connection for mobile devices, are the same with the VR parts on PC. We will discuss the database connection in later sections.

In general, we have manufactured an integrated AR system which satisfies our engineering requirements. Vuforia and software components are free. Considering the fact that most users have mobile devices on their own, our total budget for the VR part is zero.

6.3 Socket library compilation

Socket library connection utilizes the free library of windows socket APIs, thus incurring no cost. The is attached in the appendix, for interested readers to experiment with. The running steps are also specified in section 6.3.3 and Fig. 6-5b.

Materials	Manufacturing process	Manufacturing Status
winsock2.h	Programming with API's	Complete
Total budget	0¥	

Table. 6-3 Material setup for socket library sub-function

6.4 Database connection and authentication

The following are the software materials we used for completing the database connection and authentication.

Materials	Manufacturing process	Manufacturing Status
MySQL	Write sql scripts	Complete
Huaweicloud	Deploy as the server	Complete
Public IP authentication	Configure Huaweiicloud	Complete
MySQL connector (VR)	Use Dynamic link library	Complete
MyPhpDomain (AR)	Configure Huaweiicloud	Complete
WebRequest Package (AR)	Script in Unity	Complete
Total budget	175¥/month	

Table. 6-4 Manufacturing setup for database and authentication

Notice that we have a different setting on the client ends for VR and AR, since the mobile device cannot support raw MySQL connector due to vulnerable security settings. We have completed all the manufacturing processes and conducted several tests on our connection. The budget is about 175 yuan a month which comes from the Huawei cloud service.

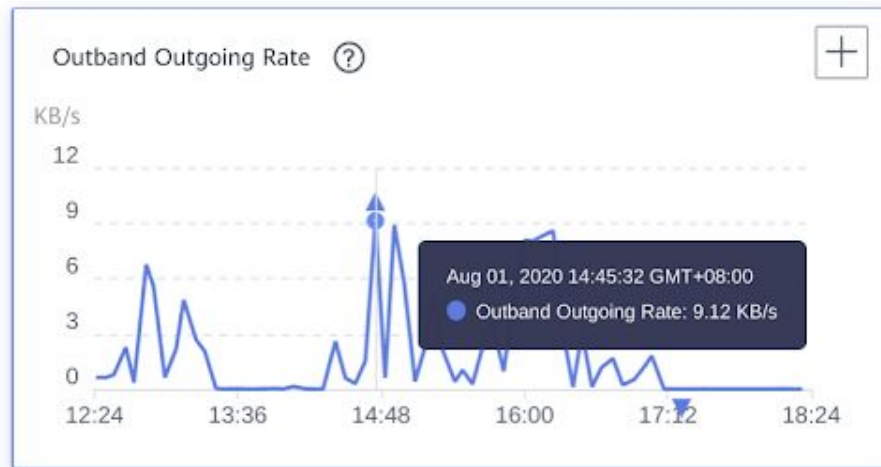


Fig. 6-4 Server Information and connection tests sample on huaweicloud

6.5 CNC Machine setup

1. CNC Milling Machine setup

The CNC Milling Machine is composed of two parts: milling machine and control box. They are connected via the cable of drilling (main axis), x/y/z axes, and position limiting steppers (see Fig. 6-6).

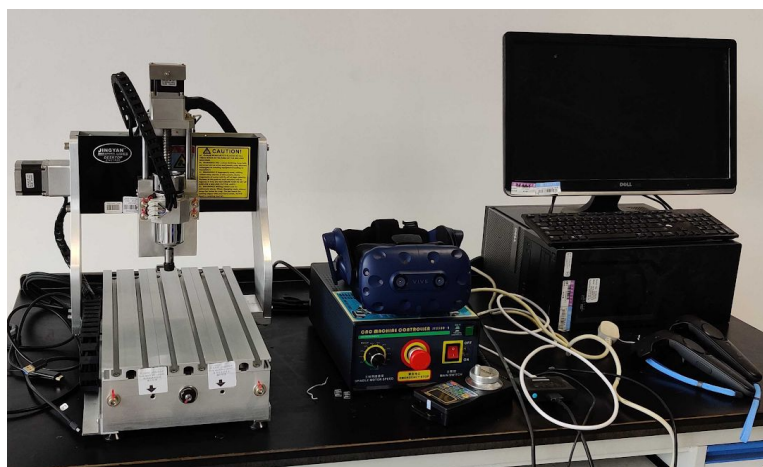


Fig. 6-5 CNC Milling Machine setup

As shown in Fig. 6-6, The CNC machine control box is connected to a computer, which is mandatorily required to be Windows 7 32-bit Operating System.

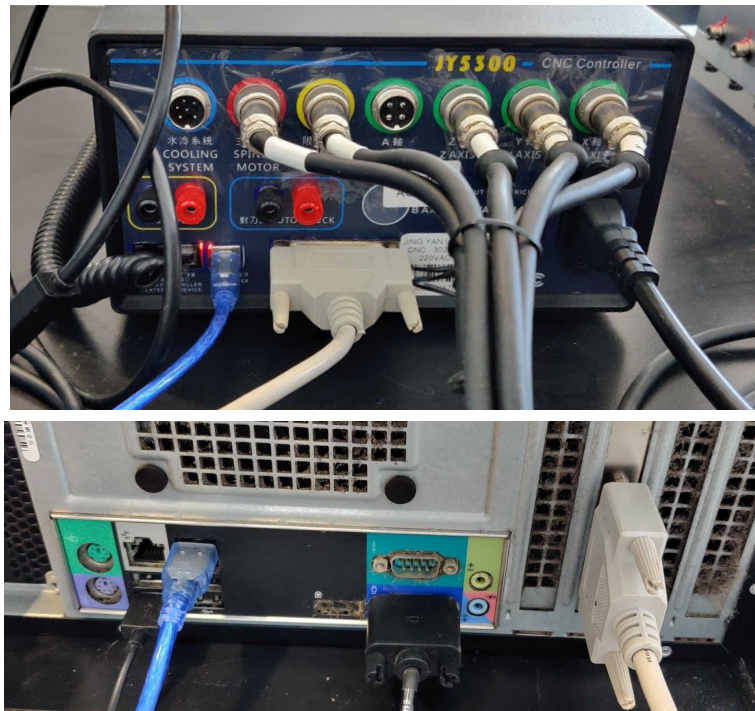


Fig. 6-6 Cables to connect the machine, the desktop, and the control box

To connect the milling machine to Windows 7 32-bit Operating System, a 25-pin PCI-e parallel card is required, as well as a USB port. Then we need to fill in the corresponding address to Mach3 software, which is specified in the “Properties” window of “LPT”. The technical parameters of this Milling machine is shown in Appendix 13.5.

The financial cost of machine setup is shown below in Table. 6-5, which is the most significant part of our budget. In addition, some cutting materials for experiments are also added to the total cost.

Materials	Manufacturing process	Manufacturing Status
CNC Milling Machine & Control Box	Assemble	Complete
PCI 25-Pin Parallel port	Install to the PCI-e socket	Complete
Materials for cutting experiments (wooden cubes, boards, organic glass)	Fixed to the milling surface	Complete
Total budget	5,700¥	

Table. 6-5 Manufacturing setup for database and authentication

2. Mach3 control

We write g-code on Mach3 to control the milling machine. The Gcode we used was generated by [jscut](#), which could turn vector files into machine codes [10]. The Gcode program can be automated by running in VB script (see Appendix 13.2).

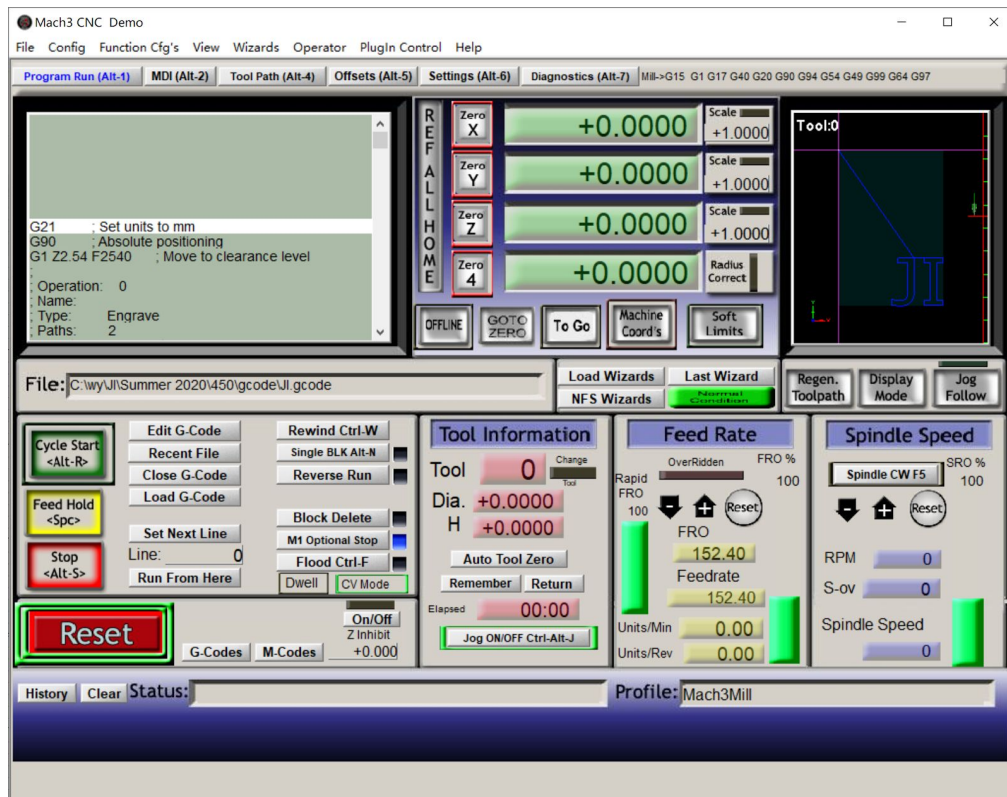


Fig. 6-8 Mach3 Interface for engraving a hollow “JI” via Milling Machine

Chapter 7 Test Results

7.1 Validate update frequency and latency

This validation is designed for the two most important engineering specifications: update frequency and latency, which as mentioned earlier is the core of user experience. Therefore, rigorous experiments must be deployed to evidence the major achievements.

Since these two variables are positively correlated (higher frequency induces higher latency), we treat them as a whole to validate. The update frequency is actively decided by MySQL script. In other words, update frequency is an independent variable to tune, while the latency is a dependent variable to measure.

To perform the test, we sampled the update frequency by tuning this parameter in MySQL database script, and record the consequent latency by network analysis tool *Wireshark*[8]. *Wireshark*[8] is one of the world's foremost and widely-used network protocol analyzers. In each TCP session, the time difference between SYN and ACK will be calculated as the round-time-trip (RTT). For example, the graph below filters and plots the RTT from address *124.71.151.195* to *10.167.124.5*, which is the traffic direction from the server to Unity. It records the time series of round-time trip time and automatically displays the average RTT, approximately 48ms, which satisfied our engineering requirement. It can be obtained by going to the (Statistics->TCP Stream Graphs->Round Trip Time) in the menu bar and selecting the stream of interest [9].

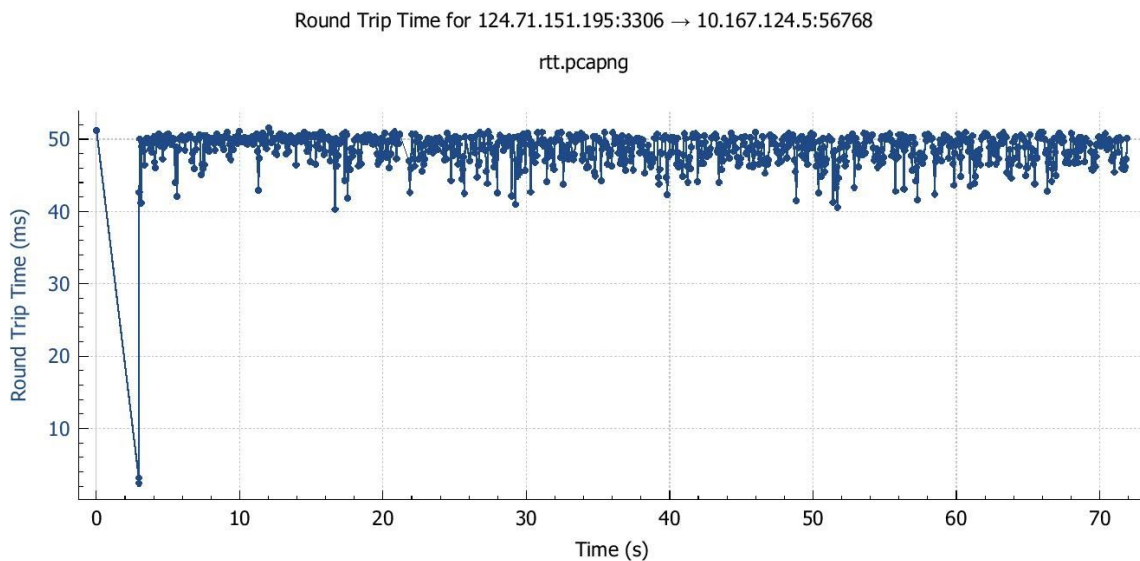


Fig. 7-1 The average round trip for filtered packets by Wireshark

For each sampled update frequency, we record the frequency-latency pairs on the

latency-frequency scatter plot for final validations.

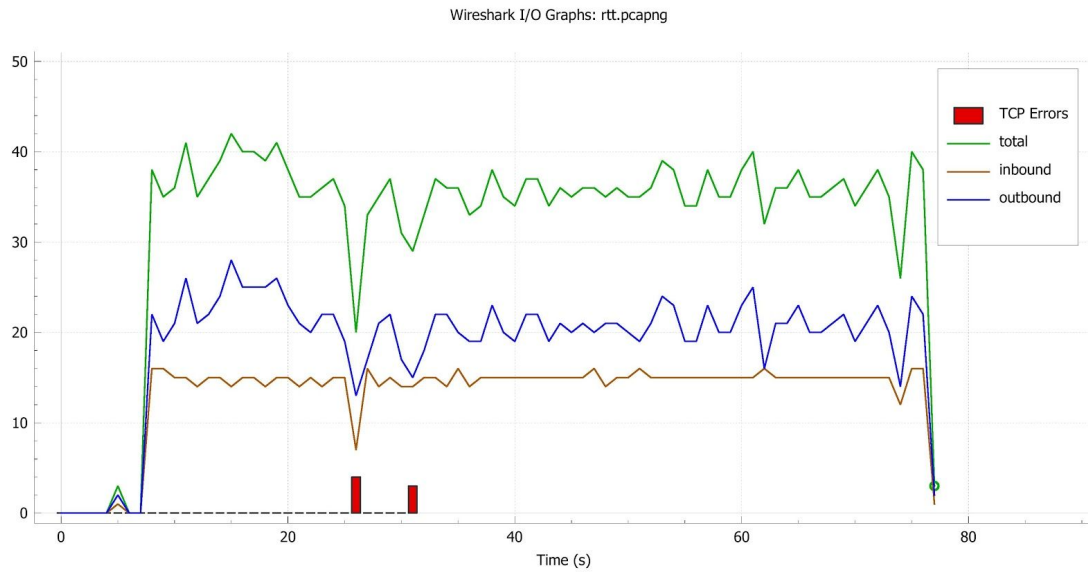


Fig. 7-2 The average inbound (brown), outbound (blue), total (green), and TCP Errors (red) packets by Wireshark

Following the previous steps, we obtain such a scatter plot, based on which we expect to find points lying in the green region, to capture high frequency but at the same time, low latency. If all the scatters or the entire fit curve lies outside of the valid region, then we claim that the system we build fails our validation. Possible solutions may include enhancing network settings, upgrading computer hardware resources (memory), or even improving Unity model algorithm, etc. Fortunately, we obtain such configurations we expected.

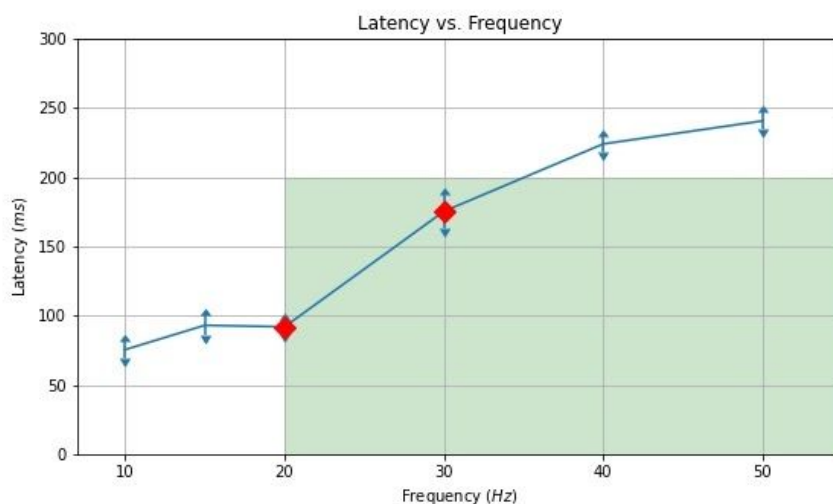


Fig. 7-3 Scatter plot generated from the experiment data, which can be used to validate update frequency and latency lying in the green valid region

7.2 Validate deviation rate

Deviation rate is the most important variable to evaluate the simulation precision. We expect the simulated virtual object to be as close to the real workpiece as possible. One of the strategies is to rasterize both the manufactured object and the simulated surface, thus each pixel can be located on the mesh grid. Then we can calculate the Euclidean distance between pairs of pixels which correspond to each other. However, to simplify the calculation process, we compared the area of the carved regions. The rate of deviation is $\sim 11\%$, which is a little higher than 7% . This demonstrates the trade-off between accuracy and efficiency, since higher resolution of carving requires more points thus much more computational power for Unity, which goes beyond our budget.

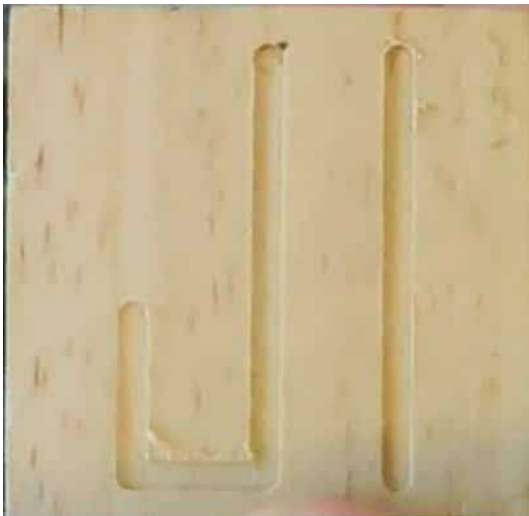


Fig. 6-15a A photo of a real workpiece



Fig. 6-15a A screenshot of a virtual workpiece in AR

7.3 Variables that are not validated

The neglected specifications in the validation plan, including Field of View, Financial cost, Handle control, and the number of supported operating systems, are either decided by precedent hardware selection, or labeling clearly on product specifications. For example, the Field of View parameter of a VR headset is a quite important factor specified on the product manual, thus it's redundant to repeat the procedure of validating such variables.

Chapter 8 Engineering Changes Notice

1. Change of server choice

Since Design review 3, we performed validation tests on system latency. However, we found that the latency is around 400~500ms, while the update frequency is limited to 3~5Hz, which is much lower than our expectations. After analysis, we gave several possible causes: campus network, limited calculation power of win7-32bit, the distance of Google server. Since the last one is much easier to improve, we redeployed our MySQL server on Huaweicloud, whose host server is located in Shanghai. After this change, the latency is reduced to ~48ms, with an update frequency of 20 packets/s. This change was made on July 27.

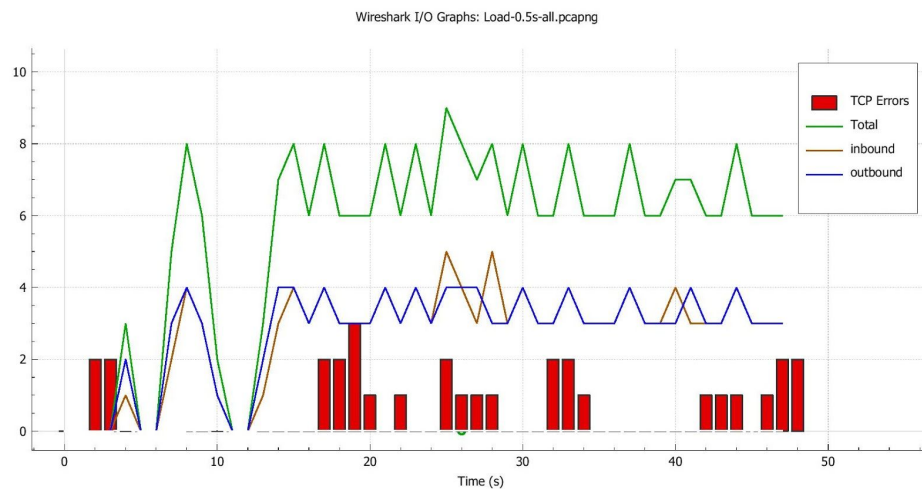


Fig. 8-1a IO graph with Google Cloud

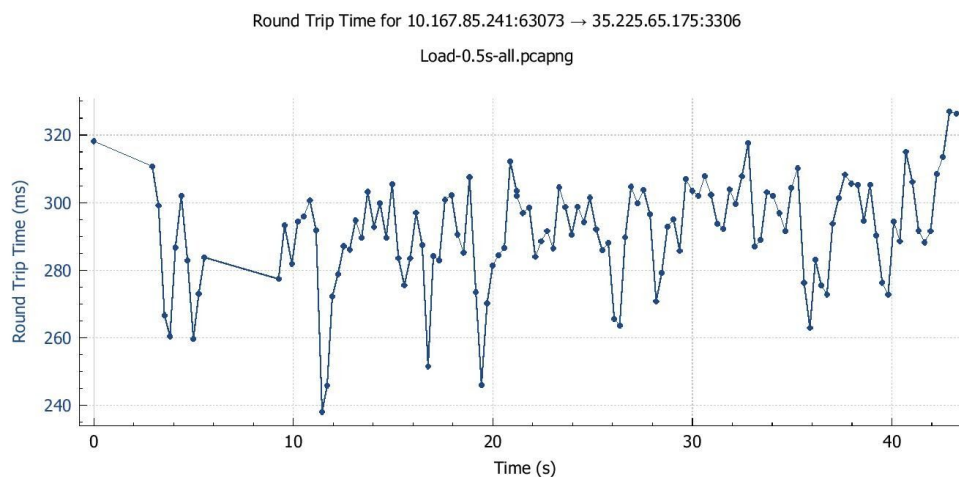


Fig. 8-1b RTT graph with Google Cloud

Chapter 9 Discussions

9.1 Strengths and weaknesses

9.1.1 Strengths

1. One of the greatest strengths, as we always emphasized, is to enable remote access to manufacturing machines, with smooth and real-time experience.
2. The database storage makes the review of the manufacturing process possible. Users can always replay the manufacturing process of some Gcode program for demonstration, modification, etc.
3. The workpiece simulation algorithm is the best innovation within this project. By transforming the object surface to point mesh and dynamically adjust the point height presents an excellent effect of carving, and maintains low computational complexity at the same time. In terms of Augmented Reality, it saves time and manufacturing materials, as the simulation avoids the real carving process. In addition, it provides a higher degree of safety than starting a Gcode program on a real machine without validation.

9.1.2 Weaknesses

1. The hardware settings of our testing machine is one of the minor weaknesses. Since the CNC Milling Machine only supports Windows7-32bit, the memory capacity and computation power is limited. However, we believe our integrated design of this system is of high compatibility, which can take effect on the improvement of each component without interfering with each other.

9.2 Potential improvements

9.2.1 Computer hardware resource requirements

Currently, the hardware of win7 32-bit that we are experimenting with is quite limited in performance. Apart from network connection, the computer performance greatly decides the response speed of the Mach3 parameter calculations, which leads to the mitigation of the final delivery of the entire system. For the future improvements, upgrading the computer hardware would be a feasible approach to enhance user experience.

9.2.2 The impact of camera resolution on Augmented Reality function

The Vuforia package that we used in Augmented Reality function is compatible with almost all platforms, but different cameras with various resolutions and computational power would give different performance. For example, our AR tests on iPhone, iPad and Xiaomi obtain quite good rendering effects, while tests on other Android systems with higher camera resolutions would incur notable delays on image tracking. In other words, the 3D model of the workpiece would be shifted on the screen and cannot keep pace with rapid, continuous change of view of the camera. This should be a built-in issue of the *Vuforia* package for some of the smart phones, but its convenience and compatibility with environments outperforms other AR libraries for Unity.

Chapter 10 Recommendations

In this project, we have implemented the and synchronized the real manufacturing and digital simulation. Virtual/Augmented Reality are both great for remote controls and feedback with high interactivity. Its compatibility and flexibility makes future improvements easier to start.

The recommendations for future developers is that one should carefully consider the hardware settings of the computer controlling CNC Milling machine. The control system of CNC Milling Machine requires an old Operating System and 25-pin cable, which greatly limits the possible speed of parameter calculating. Thus, to further improve on the performance of edge computing, upgrading the hardware is a good start.

To develop more on the control system, one has to write their own system call functions, wraps them to libraries and integrates the libraries to VB script, the only supported script language by Mach3. Therefore, one needs to pay attention to the type cast between different languages and carefully manage the memory resources.

Also, a careful choice of server location is necessary. The geo-location of the server, as mentioned in the prior sections, tremendously affects the final delivery of our project. For future developments, one could take into consideration their scope of project, and select appropriate server location.

In terms of Augmented Reality, one should examine the screen parameters like resolution, to customize the user experience for each operating environment.

In addition to engineering choices, one should manage their time in detail. During this pandemic, access to campus is much more difficult than normal. Without careful planning, it's likely that no progress is made for an entire weekend because, for example, some of the teammates might keep changing their weekly plan or are busy dealing with unexpected situations (computer crash, etc).

Chapter 11 Conclusions

In this project of VR/AR control of CNC milling machines project, we have completed the conceptual development stage and are on the right track toward the implementation stage. We have researched key concepts and compared different benchmarks. We have studied the expected outcome and the customer requirements. We have settled engineering specifications and completed QFD analysis. We have divided team roles and formed a clear schedule. We have implemented part of the CNC real time control system, including a database, a raw Unity model for the milling machine and scripts for executing machine commands remotely. We have also installed and configured hardware for VR/AR input and set up the VR/AR software development environment. At last we sped up implementation of CNC real time control and Design of AR system, as well as proved that our system met the customer requirement and engineering specifications.

We implemented a precise data reading mechanism via writing VisualBasic scripts on Mach3; We obtained full control of the milling machine by running g-codes; We built a 3D model of the milling machine in Unity; we developed a shader in Unity to simulate the articulation of artifact in VR/AR; we implemented real-time transmission by programming socket library; we maintained a MySQL database and deployed a stable server on huaweicloud platform. After a careful choice of geo-location, the latency of server communication is reduced to our validation range. We built static address websites (PHP) to empower database connection on mobile devices. The VR environment on HTC Vive pro is connected by Unity 3D, in which a CNC Milling machine model will synchronize the real manufacturing process. For Augmented Reality function, we designed specific tags with distinct features to strengthen detection and facilitate localization. In addition, we developed IOS/Android AR apps on top of vuforia toolkits so that a virtual workpiece is processed according to a given Gcode, which serves as an industrial strategy to avoid wasting materials and raising safety level.

In summary, we developed our project using selected concepts and strictly followed the engineering requirements, within the budget expectation. Our final delivery has fulfilled the expected functionality requested by our sponsor, but we can still foresee sustainable improvements to be developed from our system.

Chapter 12 Acknowledgements

Throughout this semester, our team received much help from our sponsor, instructor, and former members of related projects. We cannot achieve this without their generous support.

First we want to thank Professor Mian Li, our sponsor and at the same time our instructor. He closely supervised our progress, supported what we asked for (hardware and technical support), helped reach out to the former members of prior projects, and gave concise but detailed revision suggestions. He guided us through and guaranteed the quality of our work.

Also we want to thank Ph.D. candidates Mr. Zongyang Hu, Mr. Chenzhi Zhang, and Mr. Tianyu Wang from UM-SJTU Joint Institute. They gave us full support and helped a lot with our hardware configurations when we were off campus during this pandemic. They also provided us with precious advice on the 3D Unity shader algorithms. It's unimaginable how much effort we would put into the wrong directions without their assistance.

References

- [1] M. H. Sreekanta, A. Sarode and K. George, "Error Detection using Augmented Reality in the Subtractive Manufacturing Process," 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2020, pp. 0592-0597, doi: 10.1109/CCWC47524.2020.9031141.
- [2] Lesniak, Kevin, and Conrad S. Tucker. "Real-Time Occlusion Between Real and Digital Objects in Augmented Reality." ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers Digital Collection, 2018.
- [3] C. Perkins, 2019 augmented and virtual reality survey report
<https://www.perkinscoie.com/images/content/2/1/v4/218679/2019-VR-AR-Survey-Digital-v1.pdf>
- [4] Petrov, Christo. "35 Virtual Reality Statistics That Will Rock the Market in 2020"
<https://techjury.net/blog/virtual-reality-statistics/#gref>, accessed on June 8, 2020
- [5] Luo, W., Hu, T., Zhang, C. et al. Digital twin for CNC machine tool: modeling and using strategy. J Ambient Intell Human Comput 10, 1129–1140 (2019). <https://doi.org/10.1007/s12652-018-0946-5>
- [6] Fab Lab Connection official website <https://www.fablabconnection.com/>
- [7] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., ... & Vogels, W. (2007). Dynamo: amazon's highly available key-value store. ACM SIGOPS operating systems review, 41(6), 205-220.
- [8] Wireshark official website, <https://www.wireshark.org/>
- [9] Yanko Sosa, Calculating Average Round Trip Time of a TCP Stream
<https://www.micrium.com/calculating-average-round-trip-time-of-a-tcp-stream/>
- [10] GCode program generation tool, jscut official website <http://jscut.org/jscut.html>
- [11] Vuforia Unity 3D Developer official website <https://developer.vuforia.com/>

Chapter 13 Appendices

13.1 Socket library server-side code in C

```
// gcc -c -DBUILD_DLL server.c
// gcc -shared -o libsocket.dll server.o -Wl,--add-stdcall-alias -lws2_32
-lws2_32
#ifdef BUILD_DLL
#define EXPORT __declspec(dllexport)
#else
#define EXPORT __declspec(dllimport)
#endif

#undef UNICODE
#define _WIN32_WINNT 0x501
#ifndef WIN32_LEAN_AND_MEAN
#define WIN32_LEAN_AND_MEAN
#endif

#include <windows.h>
#include <winsock2.h>
#include <ws2tcpip.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

#define DEFAULT_BUFLen 512

SOCKET ListenSocket = INVALID_SOCKET;
SOCKET ClientSocket = INVALID_SOCKET;

EXPORT int __stdcall initServer(char *port) {
    WSADATA wsaData;
    int iResult;

    struct addrinfo *result = NULL;
    struct addrinfo hints;

    // Initialize Winsock
    iResult = WSASStartup(MAKEWORD(2,2), &wsaData);
    if (iResult != 0) {
        printf("WSAStartup failed with error: %d\n", iResult);
        return -1;
    }

    ZeroMemory(&hints, sizeof(hints));
    hints.ai_family = AF_INET;
    hints.ai_socktype = SOCK_STREAM;
    hints.ai_protocol = IPPROTO_TCP;
    hints.ai_flags = AI_PASSIVE;

    // Resolve the server address and port
    iResult = getaddrinfo(NULL, port, &hints, &result);
    if (iResult != 0) {
        printf("getaddrinfo failed with error: %d\n", iResult);
        WSACleanup();
        return -1;
    }

    // Create a SOCKET for connecting to server
```

```
ListenSocket = socket(result->ai_family, result->ai_socktype,
result->ai_protocol);
if (ListenSocket == INVALID_SOCKET) {
    printf("socket failed with error: %ld\n", WSAGetLastError());
    freeaddrinfo(result);
    WSACleanup();
    return -1;
}

// Setup the TCP listening socket
iResult = bind( ListenSocket, result->ai_addr, (int)result->ai_addrlen);
if (iResult == SOCKET_ERROR) {
    printf("bind failed with error: %d\n", WSAGetLastError());
    freeaddrinfo(result);
    closesocket(ListenSocket);
    WSACleanup();
    return -1;
}

freeaddrinfo(result);

iResult = listen(ListenSocket, SOMAXCONN);
if (iResult == SOCKET_ERROR) {
    printf("listen failed with error: %d\n", WSAGetLastError());
    closesocket(ListenSocket);
    WSACleanup();
    return -1;
}

// Accept a client socket
ClientSocket = accept(ListenSocket, NULL, NULL);
if (ClientSocket == INVALID_SOCKET) {
    printf("accept failed with error: %d\n", WSAGetLastError());
    closesocket(ListenSocket);
    WSACleanup();
    return -1;
}

// No longer need server socket
closesocket(ListenSocket);
return 0;
}

EXPORT int __stdcall sockRead(char *buff) {
    int iResult;
    iResult = recv(ClientSocket, buff, DEFAULT_BUFLen, 0);
    if (iResult > 0) {
        printf("Bytes received: %d\n", iResult);
    }
    else if (iResult == 0)
        printf("Connection closing...\n");
    else {
        printf("recv failed with error: %d\n", WSAGetLastError());
        closesocket(ClientSocket);
        WSACleanup();
        return 0;
    }
    return iResult;
}

EXPORT int __stdcall sockWrite(const char *buff) {
    int iSendResult = send(ClientSocket, buff, strlen(buff), 0);
    if (iSendResult == SOCKET_ERROR) {
```

```

        printf("send failed with error: %d\n", WSAGetLastError());
        closesocket(ClientSocket);
        WSACleanup();
        return 0;
    }
    printf("Bytes sent: %d\n", iSendResult);
    return iSendResult;
}

EXPORT int __stdcall endServer() {
    // shutdown the connection since we're done
    int iResult = shutdown(ClientSocket, SD_SEND);
    if (iResult == SOCKET_ERROR) {
        printf("shutdown failed with error: %d\n", WSAGetLastError());
        closesocket(ClientSocket);
        WSACleanup();
        return -1;
    }

    // cleanup
    closesocket(ClientSocket);
    WSACleanup();
    return 0;
}

```

13.2 Mach3 control and communication code in VB

```

Dim res As Integer
Dim buff As String
Dim IsStart As Boolean
IsStart = false
buff = Space$(512)

Public Declare Function initServer Lib "libsocket.dll" _
    (ByVal port As Integer) As Integer
Public Declare Function endServer Lib "libsocket.dll" _
    () As Integer
Public Declare Function sockRead Lib "libsocket.dll" _
    (ByVal buff As String) As Integer
Public Declare Function sockWrite Lib "libsocket.dll" _
    (ByVal buff As String) As Integer

res = initServer("8080")

Open "C:\wy\JI\Summer 2020\450\gcode\test.txt" For Append As #1

Do While True
    res = sockRead(buff)
    message res
    If res > 0 Then
        buff = Left$(buff, res)
        sockWrite(buff)
        Print #1, buff

        Open "C:\wy\JI\Summer 2020\450\gcode\record.txt" For Output As #2

        x = Round(GetParam("XMachine"), 2)
        y = Round(GetParam("YMachine"), 2)
        z = Round(GetParam("ZMachine"), 2)
        vx = Round(GetParam("VelocitiesX"), 2)
        vy = Round(GetParam("VelocitiesY"), 2)
        vz = Round(GetParam("VelocitiesZ"), 2)
    End If

```



```
Print #2, x, " ", y, " ", z, " ", vx, " ", vy, " ", vz
Close #2
message "record"

If buff = "load" Then
    LoadFile("C:\wy\JI\Summer 2020\450\gcode\gtest1.nc")
    message "load"
End If

If buff = "start" Then
    RunFile()
    message "runfile"
End If

' 1021 reset, 1000 cyclestart, 1001 feed hold, 1002 rewind, 1003 stop
If buff = "pause" Then
    DoOEMButton(1001)
    ' 1003 for immediate stop
    message "kill"
End If

If buff = "continue" Then
    DoOEMButton(1000)
    message "continue"
End If

If buff = "rewind" Then
    DoOEMButton(1002)
    message "rewind"
End If

If buff = "reset" Then
    DoOEMButton(169)
    message "reset"
End If

If buff = "stop" Then
    DoOEMButton(1021)
    sleep(1000)
    DoOEMButton(1021)
    sleep(500)
    DoButton( 24 )
    DoButton( 23 )
    DoButton( 22 )
    DoButton( 25 )

    DoOEMButton(133)
    DoOEMButton(134)
    DoOEMButton(135)
    sleep(1000)
    DoOEMButton(1000)
    message "stop"
End If

If Left(Mid(buff, 1), 1) = "G" Then
    Code(buff)
End If
End If
Loop

res = endServer()
```

13.3 Partial Unity 3D Model Shader code in C#

13.3.1 CylinderCut.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CylinderCut : MonoBehaviour
{
    public float moveSpeed = 2.0f;

    Vector3 centerOffset;
    float height;

    // Start is called before the first frame update
    void Start()
    {
        centerOffset = GetComponent<CapsuleCollider>().center;
        height = GetComponent<CapsuleCollider>().height *
gameObject.transform.localScale.y / 2;
    }

    // Update is called once per frame
    void Update()
    {
        // float MoveH = Input.GetAxis("Horizontal");
        // float MoveV = Input.GetAxis("Vertical");
        // float MoveZ = Input.GetAxis("Mouse ScrollWheel") * 3.0f;
        // transform.Translate(new Vector3(MoveH, MoveZ, MoveV) * Time.deltaTime
* moveSpeed);

        EventBus.Publish<HeightEvent>(new
HeightEvent(gameObject.transform.position.y - height));
    }
}

public class HeightEvent{
    public float height;
    public HeightEvent(float _height)
    {
        height = _height;
    }
}
```

13.3.2 VertexGenerator.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.Text;
using System;

public class VertexGenerator : MonoBehaviour
{
    //控制点的大小
    float pointScale = 0.01f;
```

```
public GameObject editorpoint;

Mesh mesh;

//顶点列表
List<Vector3> positionList = new List<Vector3>();
//顶点控制物体列表
List<GameObject> positionObjList = new List<GameObject>();

/// <summary>
/// key:顶点字符串
/// value:顶点在列表中的位置
/// </summary>
Dictionary<string, List<int>> pointmap = new Dictionary<string, List<int>>();

// 顶点翻倍次数
int iteration = 5;
Vector3[] origVertices;
int[] origTriangles;

void Awake()
{
    mesh = transform.GetComponent<MeshFilter>().mesh;
    for (int i = 0; i < iteration; i++)
    {
        IncreseVertices();
    }
}

// Start is called before the first frame update
void Start()
{
    mesh = GetComponent<MeshFilter>().sharedMesh;
    CreateEditorPoint();
}

// Update is called once per frame
void Update()
{
}

//创建控制点
public void CreateEditorPoint() {
    positionList = new List<Vector3>(mesh.vertices);
    for (int i = 0; i < mesh.vertices.Length; i++)
    {
        if (mesh.vertices[i].y != 0.5)
        {
            continue;
        }
        string vstr = Vector2String(mesh.vertices[i]);
        if (!pointmap.ContainsKey(vstr)) {
            pointmap.Add(vstr, new List<int>());
        }
        pointmap[vstr].Add(i);
    }
    foreach (string key in pointmap.Keys)
    {
        editorpoint = Instantiate(editorpoint);
        editorpoint.transform.parent = transform;
        editorpoint.transform.localPosition = String2Vector(key);
    }
}
```

```

        editorpoint.transform.localScale = new Vector3(pointScale, pointScale,
pointScale);
        MeshEditorPoint editorPoint =
editorpoint.GetComponent<MeshEditorPoint>();
        editorPoint.onMove = PointMove;
        editorPoint.pointid = key;
        positionObjList.Add(editorpoint);
    }
}
//顶点物体被移动时调用此方法
public void PointMove(string pointid, Vector3 position) {
    Debug.Log("Here");
    if (!pointmap.ContainsKey(pointid)) {
        return;
    }
    List<int> _list = pointmap[pointid];
    for (int i = 0; i < _list.Count; i++) {
        positionList[_list[i]] = position;
    }
    mesh.vertices = positionList.ToArray();
    mesh.RecalculateNormals();
}

string Vector2String(Vector3 v) {
    StringBuilder str = new StringBuilder();
    str.Append(v.x).Append(", ").Append(v.y).Append(", ").Append(v.z);
    return str.ToString();
}
Vector3 String2Vector(string vstr)
{
    try {
        string[] strings = vstr.Split(',');
        return new Vector3(float.Parse(strings[0]), float.Parse(strings[1]),
float.Parse(strings[2]));
    } catch (Exception e) {
        Debug.LogError(e.ToString());
        return Vector3.zero;
    }
}

void IncreaseVertices()
{
    if (mesh.vertexCount >= 18432)
    {
        Debug.Log("Too Many");
        return;
    }
    origVertices = mesh.vertices;
    origTriangles = mesh.triangles;
    Dictionary<Vector3, int> verticesResultDic = new Dictionary<Vector3, int>();
    List<int> trianglesResultList = new List<int>();
    //计算三角面的个数
    int k = origTriangles.Length / 3;
    int index = 0;
    for (int i = 0; i < k; i++)
    {
        //取出一个三角面 (的顶点)
        Vector3[] triangle = new Vector3[3] { origVertices[origTriangles[i * 3]],
origVertices[origTriangles[i * 3 + 1]], origVertices[origTriangles[i * 3 + 2]] };
        //通过取三条边的中心点
        //原来三个顶点, 变成六个顶点
        Vector3[] result = new Vector3[6];
        Vector3 v01 = (triangle[0] + triangle[1]) * 0.5f;
        Vector3 v12 = (triangle[1] + triangle[2]) * 0.5f;
    }
}

```

```

Vector3 v02 = (triangle[0] + triangle[2]) * 0.5f;
if (AddVertices(verticesResultDic, triangle[0], index)) index++;
if (AddVertices(verticesResultDic, triangle[1], index)) index++;
if (AddVertices(verticesResultDic, triangle[2], index)) index++;
if (AddVertices(verticesResultDic, v01, index)) index++;
if (AddVertices(verticesResultDic, v12, index)) index++;
if (AddVertices(verticesResultDic, v02, index)) index++;
// 将原三角面分成新的四个三角面
// 注意左手法则, 逆时针顺序
// 三角形数组存储的是顶点在顶点数组中的序号
trianglesResultList.Add(verticesResultDic[triangle[0]]);
trianglesResultList.Add(verticesResultDic[v01]);
trianglesResultList.Add(verticesResultDic[v02]);
trianglesResultList.Add(verticesResultDic[v01]);
trianglesResultList.Add(verticesResultDic[triangle[1]]);
trianglesResultList.Add(verticesResultDic[v12]);
trianglesResultList.Add(verticesResultDic[triangle[2]]);
trianglesResultList.Add(verticesResultDic[v02]);
trianglesResultList.Add(verticesResultDic[v12]);
trianglesResultList.Add(verticesResultDic[v02]);
trianglesResultList.Add(verticesResultDic[v01]);
trianglesResultList.Add(verticesResultDic[v12]);
}

mesh.vertices = GetReusltVertices(verticesResultDic);
mesh.triangles = trianglesResultList.ToArray();
mesh.RecalculateBounds();
// 由于normal没有增加, 导致表面看起来不平滑 (如果要重新计算normals参考顶点的计算)
mesh.RecalculateNormals();

}

bool AddVertices(Dictionary<Vector3, int> verticesResultDic, Vector3 vertice,
int index)
{
    if (verticesResultDic.ContainsValue(index) ||
verticesResultDic.ContainsKey(vertice))
        return false;
    verticesResultDic.Add(vertice, index);
    return true;
}

Vector3[] GetReusltVertices( Dictionary< Vector3,int> verticesResultDic)
{
    int length = verticesResultDic.Keys.Count;
    Vector3[] result = new Vector3[length];
    List<Vector3> temp = new List<Vector3>(verticesResultDic.Keys);
    for (int i = 0; i < length; i++)
    {
        result[i] = temp[i];
    }
    return result;
}
}

```

13.4 Partial Database connection code in PHP

```

<?php
$servername = "****";
$username = "****";
$password = "****";
$dbname = "****";
$id = $_GET["id"];

```

```
$conn = new mysqli(null, $username, $password, $dbname, null, $servername);

if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT x, y, z FROM positions WHERE id = $id";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        echo $row["x"]. ", " . $row["y"]. ", " . $row["z"];
    }
} else {
    echo "****";
}
$conn->close();
```

13.5 CNC Milling Machine Parameters

产品型号	3020-300/800W	4030-300/800W/1.5/2.2KW	3040-1.5/2.2KW	4060-1.5/2.2KW
工作电压	220V/110V(需定制)			
台面尺寸	240x440mm	540x400mm	450x480	750x500
有效行程	300x200x100mm	400x300x100	300x400x100	400x600x100
外形尺寸	450x550x520mm	650x550x540	560x590x540	860x650x540
主轴功率	风冷 300W/水冷 800W	风冷 300W/水冷 800W/1.5/2.2KW	水冷 1.5/2.2KW	水冷 1.5/2.2KW
主轴转速	12000/24000RPM	12000/24000RPM	24000RPM	24000RPM
夹头尺寸	1.5KW 及以下机型使用 ER11 1-7mm 夹头, 2.2KW 使用 ER20 1-13mm 夹头			
步进电机	57 型 3A	57 型 3A	57 型 3A	57 型 3A
	57 型 3A	57 型 3A	57 型 3A	57 型 3A
	57 型 3A	加长 57 型 3A	加长 57 型 3A	加长 57 型 3A
滑动单元	X 16 镀铬光轴	20 镀铬光轴	20 镀铬光轴	20 镀铬光轴
	Y 16 镀铬光轴	20 镀铬光轴	20 镀铬光轴	20 镀铬光轴
	Z 12 镀铬光轴	12 镀铬光轴	12 镀铬光轴	16 镀铬光轴
传动单元	1605 滚珠丝杆	1605 滚珠丝杆	1605 滚珠丝杆	1605 滚珠丝杆
限位开关	共六个限位开关 (3020 机型为物理限位开关, 其余机型为磁编码器限位)			
机器精度	0.02-0.05mm	0.02-0.05mm	0.02-0.05mm	0.02-0.05mm
重复精度	0.01-0.02mm	0.01-0.02mm	0.01-0.02mm	0.01-0.02mm
加工速度	0-3500mm/min			
机器重量	38/43kg	52/60/65kg	65kg	70kg
包装尺寸	49x59x60cm	68x58x62cm	68x58x62cm	89x68x62cm

Fig. 13-1 The technical parameters of CNC Milling machine

Bill of Materials

Qty.	Items	Purchased from	Type	Unit Price (RMB)
1	CNC Milling Machine & Control Box	Jieke Automation Equipments	Jingyan 3020-300W	5,500
2	PCI-e 25-Pin Parallel port	Jieke Automation Equipments	PCI + PCI-e	25
1	Mini-dp cable			139
2	Unity 3D model of CNC Milling Machine Components			10
30	Materials for cutting experiments (wooden cubes, boards, organic glass)	Yidimei, Pangxiewangguo	5*5*5cm, 4*4*4cm, 3*3*3cm	5
1	Server Maintenance fee (Huawei Cloud)	Huaweicloud.com	2vCPUs+ 4GB, c6s.large.2	175
	Total: 6,034			

Table. Total budget of this project

Bios

Yue Wu

B.S. candidate in ECE in Shanghai Jiao Tong University and B.S. in Computer Science in University of Michigan. My interest lies in autonomous robotics, especially the field of decision making and visual perception. I've been working on topics like multi-policy decision making, learning object articulation of objects and human model reconstruction from videos. I would devote myself to robotics research. As the team leader of this project, I would organize and schedule team tasks. I am also responsible for mach3 scripting, 3D modeling and VR/AR development.



Zhiqi Chen

B.S. candidate in ECE in Shanghai Jiao Tong University and B.S. in Computer Science in University of Michigan. I worked on operating system optimization, autonomous robotics and game design. My future plan is to be a researcher in the realm of operating systems or to be a game designer. In this capstone design project, I'm responsible for Unity 3D modeling, database server maintenance, and backend connectivity.



Junwei Deng

B.S. candidate in ECE in Shanghai Jiao Tong University and M.S. candidate in Information student at University of Michigan. I worked on deep learning research on biomedical image, graph neural networks on cold start problems, and high dimension TS forecasting. I worked in Intel as a deep learning software intern. In this project, I am responsible for the mach3 script and connection between the CNC machine and database, hardware machine deployment.



Yue Zhang

B.S. candidate in ECE in Shanghai Jiao Tong University and B.S. candidate in CS in University of Michigan. I worked on Game Development and Natural Language Processing. My future career plan is to work in game development, especially in the



AR/VR field. In this project, I take responsibility for construction of the AR/VR environment for the digital twin inside the virtual factory.

Tianyi Ge

ECE student at Shanghai Jiao Tong University and Information Science (Data Science) student at University of Michigan. I worked on distributed systems, AWS app development, and container scheduling projects. Impressed by *EECS 491 Introduction to Distributed Systems*, my future plan is to be a researcher in the realm of distributed cloud/edge systems. In this capstone design project I'm responsible for purchasing, database testing, network communication, and machine connection.

