Automatic Air Hockey Robot

Wang Futong, Sun Chengyuan, Xu Mingjie

***Abstract*— We designed and implemented an autonomous air hockey robot. The bot can strategize on its own and play against human players. To simplify the operation of air hockey, we also implemented gesture-based game play, where human players can control their rackets by moving their hands and changing gestures. These changes could help the medical system to help patients in need of nerve rehabilitation in the arm, and significantly reduce costs.**

This system is based on ROS and simulated using gazebo, and the core code is implemented in python. `https://github.com/Xumj82/air_hockey_robot`

***Index Terms*— Air-hockey robot, ROS bridge, Gazebo simulation, human machine interface, Device-Based Rehabilitation Therapy**

I. Introduction

A Very important part of modern medicine is the rehabilitation of injured or disabled limbs. Among these rehabilitation therapies, one approach is machine-based, allowing the patient to perform prescribed movements to stimulate reflexes and mobility. Air hockey is an emerging form of these therapies. Because of its game-based approach, it can capture the patient's attention more than conventional means and can help them perform their workouts better.

In the actual use of air hockey as a rehabilitation device, two problems are often mentioned: 1. Since this is a confrontational game, the patient must be accompanied by a nurse for rehabilitation training, which will cost a lot of money. 2. The use of air hockey can exercise arm flexibility and brain response ability very well, but in actual play, the patient's legs are often more demanding, which makes many elderly people with mobility problems unable to participate in this sport.

We offer a new way of virtual play, by digitally modeling an air hockey ball, we can allow patients to use hand gestures to control the racket without physically moving their legs. By offering a robotic opponent, patients don't have to spend more to hire a carer to accompany them on tours.

The entire system is based on full ROS and digital modeling, which means that these jobs are ready to be converted into actual robots (rather than digitally modeled) and put into use.

In our implementation, we use machine learning to recognize user gestures, computer graphics methods to identify and predict hockey ball positions and routes, and rule-based methods to design robot strategies. The whole system is based on ROS, and the simulation work is realized by gazebo.

We use some external resources to simplify the workload, but the modeling of the subject, policy formulation, gesture recognition and integration are all done by us independently.

II. Related Work

A. Human–robot interaction

Human-robot interaction is a important function when designing a robot. Various methods and technologies, such as kinesthetic-based, joy-stick-based, vision-system-based, and natural-language-based , have been developed and implemented in the Human–robot interaction.

Kinesthetic-based is also known as kinesthetic teaching, is a method to record human's action with the sensors in robot, and play a series actions by reproducing sensor messages. In this method, the trajectory shown by human will be regarded as a physical interaction with robot. This method provide a simple interface to teach the robot for complicated actions. But this type of interaction is always be implemented in the scenario which the robot is not required to react for dynamic environment changing.

Joy-stick-based, vision-system-based or natural-language-based interaction are usually more flexible comparing with kinesthetic-based. And comparing with the other two methods, joy-stick-based interaction is always more precise. In this case, joy-stick-based system also requires more for human to demonstrate more precise operations.*However, not all systems are suitable for this technique since the operated robot must be manageable without a high degree of freedom or a complicated structure* [1].

Vision-system-based and natural-language-based interaction provide a compromise approach for simple interface and precise operation, especially an AI based vision system. With the enhancement of a high-accuracy artificial intelligence model recognition system, the vision-based interaction system enables humans to control the robot in the most intuitive way, and also keeping high operation precision.

B. Air Hockey

Air Hockey Robot Project (a 3D printer hack) by Jose Julio [8]:

This project has real robot, the Path Prediction and Plan system has Robot vision system by Camera,Object detection and Trajectory prediction figure II-B. They use camera to capture bird view of table, Then detect hockey and make predict, use light line and green line to represent hockey's predicted path.

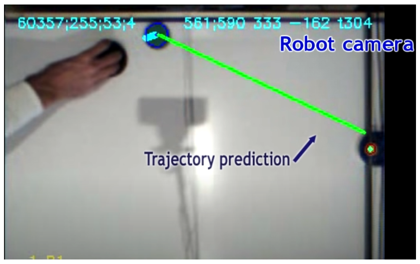

Fig. 1. Julio Air Hockey

Robot Strategy has Defense, Defense+Attack and Attack.This system will evaluate the predict status, divided into no risk, moving to our field directly and moving to our field with a bounce. Based on different predict status, they give different handle method and strategy.

Autonomous Air Hockey Robot [5]:

They use slider mechanism and a vision system using Kinect to create an Autonomous Air Hockey Robot.The project is control slide by sensing its environment of the machine leaning vision system, then play against with human figure II-B.

For the Object Detection Using Kinect They use visual detection of the hockey. Import Simple OpenNI library, PI image class for image processing and color for tracking. For the Prediction of hockey's Path they Calculate the trajectory when the hockey is detected in two consecutive frames.

Fig. 2. Autonomous Air Hockey Robot

Application of Machine Learning in Air Hockey Interactive Control System [7]:

This project, they use deep learning coupled with motor control to realize the real-time interactive system of air hockey.

For the hockey detection. they use the convolutional neural network *YOLO* [6] to capture the hockey current position,figure II-B. For the path prediction, the law of reflection and neural networking are applied to predict the end position of the puck Based on the predicted location, the system will control the stepping motor to move the linear slide to realize the real-time interactive air hockey system.

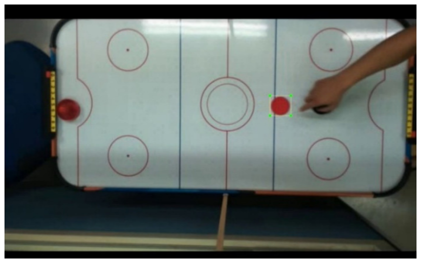

Fig. 3. Machine Learning in Air Hockey System

III. Method and System

A. Simulation and Modeling

For modeling this robot, Blender was used to build the sketch of the physical structure of the overall platform. After that, based on this sketch, more physical properties and mechanisms was defined within the urdf file.*The impact between rigid objects is a complicated phenomenon and involves such things as material deformations, friction, vibration, sound generation, and other complex dynamic effects that are either difficult to model or poorly understood* [10]. In order to reduce unnecessary calculations and restore the real scene as much as possible we set the base as static and all materials to lexan polycarbonate resin. Details of physical property of the robot are shown in table I,II and table III

In Gazebo the friction model is Coulomb friction model. It defines the relationship between the normal and tangential forces present at a contact point:

$$|f_T| <= mu * |f_N| \tag{1}$$

where f_N and f_T are the normal and tangential force vectors respectively, and mu is the friction coefficient (typically a number around 1.0). This equation defines a "friction cone" - imagine a cone with f_N as the axis and the contact point as the vertex. If the total friction force vector is within the cone then the contact is in "sticking mode", and the friction force is enough to prevent the contacting surfaces from moving with respect to each other. [2]

As friction is always unwanted in the game of air hockey, we set the mu of the hockey as 0.0001, which means only need 1N as tangential force to make the hockey move when the normal force is 10000N.

TABLE I
VISUAL PROPERTY

Part name	Width	Length	Height	Radius
Base(table)	96cm	200cm	80cm	N.A.
Tracks	N.A.	N.A.	N.A.	N.A.
Rackets	N.A.	N.A.	2cm	4cm
Hockey ball	N.A.	N.A.	1cm	4cm

TABLE II
COLLISION PROPERTY

Part name	tWidth	Length	Height	Radius	Density
Base(table)	t96cm	200cm	80cm	N.A.	N.A.
Tracks	t110cm	10cm	1cm	N.A.	1.15g/cm^3
Rackets	tN.A.	N.A.	2cm	4cm	1.15g/cm^3
Hockey ball	tN.A.	N.A.	2cm	1cm	1.15g/cm^3

TABLE III
TRANSMISSION PROPERTY

Joint name	Quantity	Hardware Interface	PID
base_to_track	2	EffortJointInterface	1000 : 0.01 : 100
track_to_racket	2	EffortJointInterface	1000 : 0.01 : 100

B. ROS Implementation

To master the robot and design a hierarchical controlling system, we must implement the topology structure on a basic OS which has abundant hardware interfaces and communication modes. So we used Robot Operating System(ROS) as the basic frame work to design the our robot system.

The overall topology structure consist 3 sub-systems as Fig.4 and Fig 5: hockey robot module, decision module and interaction module. In hockey robot module, there are 2 pairs of track node corresponding to a pair of racket, also there is a bird-view RGB camera for robot observation and a pair of ultrasonic sensors to detect goal. In decision module, there is a predictor and a planner which will be discuss later, and the other nodes are used to process message for human-robot interactions.The last module is the interaction module, which contains another camera to capture human command by gesture, and also provide a GUI to display robot response.

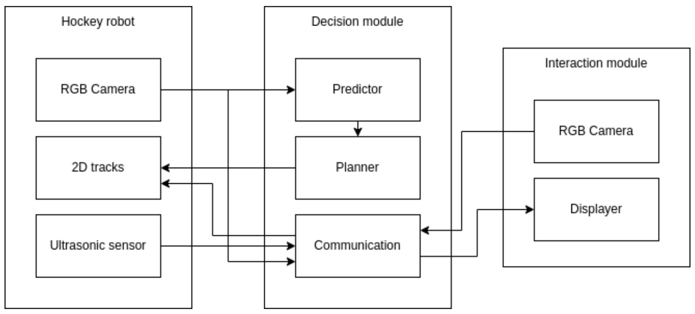

Fig. 4. System Architecture

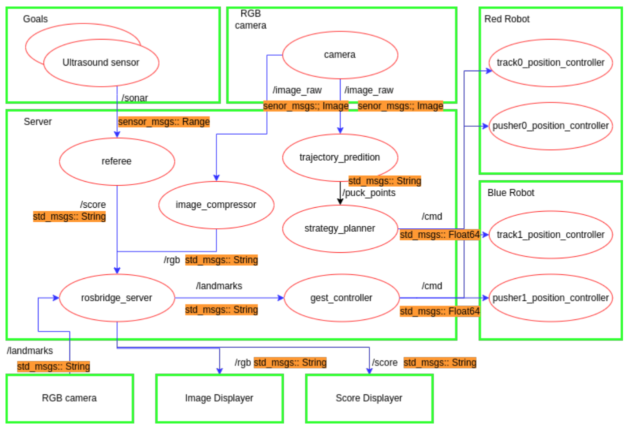

Fig. 5. ROS Architecture.There are totally 13 nodes for the whole system. 3 sensor nodes for perception, 6 nodes for processing and 4 nodes for actions.All communication between nodes are implemented by ROS topics

C. Vision-base controlling

In order to control the racket as human player, a two stages workflow as in Fig. 6 was implement for gesture controlling of the robot. At 1st stage, we use *MediaPipe* [3] to generate landmarks of one hand from in every frame captured by the user's side camera and publish those landmarks into a ROS topic. After that, at 2nd stage, the controller node will subscribe this topic and deserialize the message into landmarks, mean while, those landmarks will flatten in to a 1D array and sent to a *4 hidden layers neural network* [4] (table IV) to output the status of user's hand, which is CLOSE or OPEN,

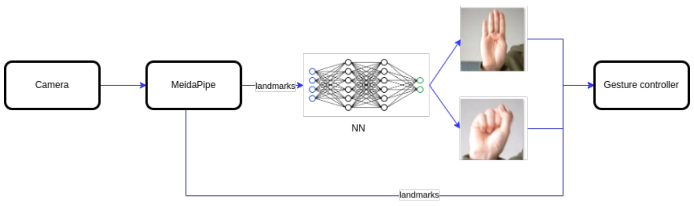

Fig. 6. Gesture control workflow. In our project, we used a JavaScript based Media-Pipe library, and the landmarks will be generate locally and transferred to a remote ROS node via web socket

TABLE IV
GESTURE RECOGNITION MODEL

Layer (type)	Output Shape	Parameter
Input	(None, 42)	0
Dropout	(None, 42)	0
Dense	(None, 20)	860
Dropout	(None, 20)	0
Dense	(None, 10)	210
Dense(output)	(None, 2)	22

The two stages are implemented by different nodes, so that the 2 parts can run with different devices which will reduce hardware requirements, especially in case the 1st stage running on client.

TABLE V
GESTURE CONTROL LOGIC

User Gesture	Simulation	Racket(robot) action
Opening	Hitting	Forwarding
Closing	Gripping	Following

Once the landmarks and gesture status are ready, controller node will decide the racket action by the fused data as table V shown. When user's hand is closing, the pixel position of the wrist point will be mapped into the half court of the hockey table, and the racket will move into the mapped point. When user's hand is opening, the racket will ignore the wrist point and move forward to simulate a hit action.

D. Path Prediction

In our modeling, we set up a bird's-eye view camera at the exact center of the table. The angle of view obtained by the camera is shown in the figure 7.

Our route predictions are based on the following assumptions:

1. Ignore the curve generated by the hockey spin due to special hitting points.

2. The collision between the hockey ball and the edge of the table is elastic, the speed is almost constant, and the direction is symmetrical along the edge normal.

The reason we make this assumption is that after reading Patrick D. Weidman's article [9] on the spin curve in hockey, we realized that the way we operate is difficult to hit such a spin. At the same time, the calculation method and calculation amount of this curve are too complicated. Treating the situation as "undefensible hockey" is the most effective option when the probability of occurrence is low and the handling is difficult.

Under this assumption, we use the HSV gamut to extract the current position of the hockey. After calculating the hockey position in two consecutive frames, we can find an approximate straight line for the hockey path.

In simple terms, this calculation can be thought of as solving the equation of a line based on the coordinates of two points.

To simplify the interaction between modules and save communication costs, we use an array of predicted positions 50 frames into the future of hockey instead of the actual route.

Since path prediction requires high real-time and accuracy, we additionally add a future multi-frame comparison strategy. The strategy calculates the hockey position for subsequent frames based on the predicted path and calculates the error from the actual camera-observed position. When the error accumulates to a threshold, we assume that the hockey line has changed for some reason. The reason for this may be due to the player making up an attack again. The solution to this error is very simple: we recalculate the path of the hockey based on the recorded frames where the error occurred.

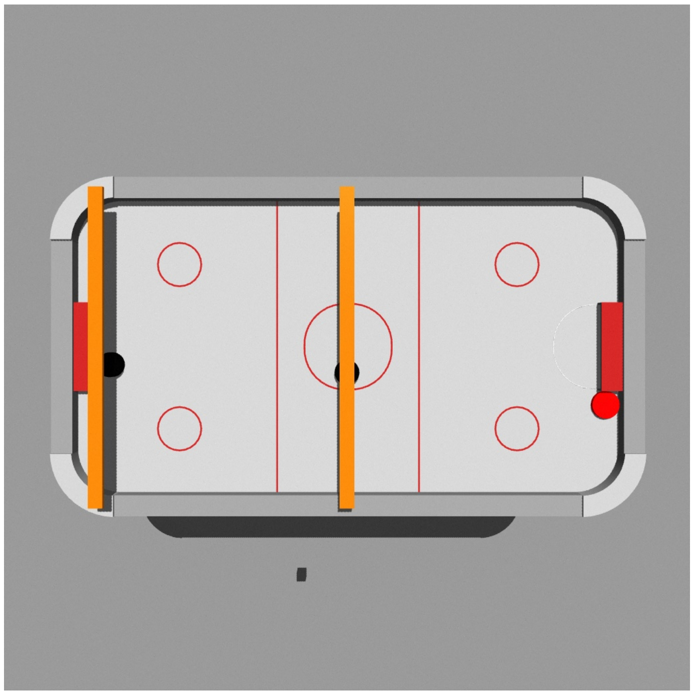

Fig. 7. Camera view

E. Strategy Plan

We have a planner to manage all Strategy. and planner will get hockey's timestamp and predicted position information from the predictor. Based on this information, we can know the trajectory from its position to the pusher, and know when the hockey will reach the robot's area. So we can control our pusher to go to the specific position according to the specified time to catch the hockey.

1) *Strategy 1: Defence:*
The robot just waited on the bottom line of the goal and adjusted the robot position to make sure it could catch the hockey, which meant the robot were just defending and not attacking.

Since we got the prediction path of hockey with timestamp as the key from predictor, we can go directly to the baseline location where hockey will reach. This will complete the catch action, which is the defense.Strategy 1 figure 8.

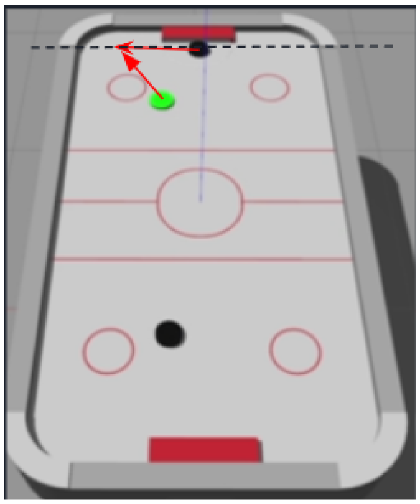

Fig. 8. Strategy 1

2) *Strategy 2: Normal Attack:*
The robot will take the initiative to attack to the predicted point, but the attack is not strong, it can only guarantee to catch the hockey. Usually used when the hockey is in our half and the game starts.

But we can also directly use strategy 2 for defense or offense. By predicting the hockey's travel route, we cyclically let the robot reach the point where each hockey is about to reach, which can guarantee the knockdown of the ball, but it is not very aggressive.Strategy 2 figure 9.

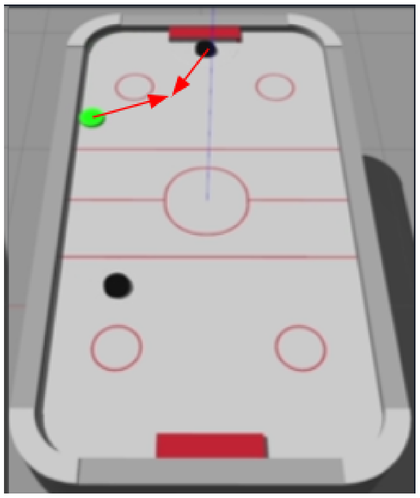

Fig. 9. Strategy 2

3) Strategy 3: Pressing Attack: Strategy 3 will be our main attack strategy, of course we also set it as the default strategy. In order to make the counterattack more aggressive, we added ATTACK LINE and SAFE LINE to the x-axis. In most cases, we let the robot wait on the ATTACK LINE, and the y-axis position is the predicted hockey position. Sure. Afterwards, when the hockey enters the attack time zone we set (the hockey is about to reach the SAFE LINE), we will launch an attack, let the robot go to the SAFE LINE, and the hitting position will be adjusted, so that the hockey returning from the counterattack always faces the opponent's goal, And adding some forward distance on the x-axis makes the hit harder and the counterattack faster. This makes the overall strategy more aggressive. The Strategy 3 algorithm can be shown :

Algorithm 1: Strategy 3

```
Set default strategy is 3;
if hockey prediction path changed then
    if hockey can reach our safe line then
        if hockey speed is too fast then
            change plan to Strategy 1;
        else
            robot go to ATTACK LINE and wait;
else
    if hockey can reach our safe line then
        if hockey speed is reach attack area then
            robot go to attack position from ATTACK
              LINE;
```

First the robot will calculate if the speed of the hockey ball will allow it to execute strategy 3, if not it will execute strategy 1. If it can: robot will make the racket wait on the ATTACK LINE and execute the attack:

$$\frac{0.86 * 2 - SL}{0.86 * 2 - SL + 0.03} = \frac{y}{AP}. \quad (2)$$

$$x = (0.86 * 2 - SL) \quad (3)$$

SL = STRATEGY SAFE LINE) , AP = strategy attack position , y = strategy defence position

So the attack position is (SL+0.05,AP), this will make us attack hockey and towards to the goal and with greater thrust. Strategy 3 figure 10.

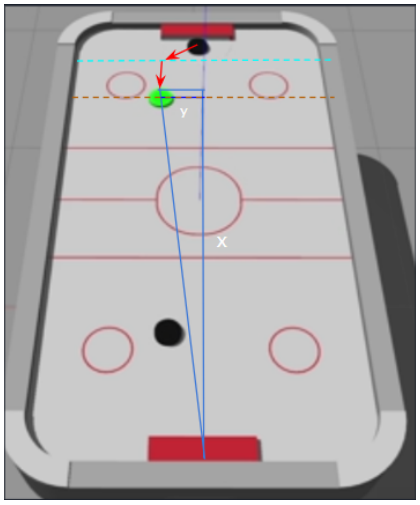

Fig. 10. Strategy 3. ATTACK LINE is light blue dotted line, SAFE LINE is brown dotted line

4) Strategy Rule base: We have created 3 strategies for defense or attack, we can decide to just use any single strategy, or all 3 strategies can be set up to be mixed throughout the game. we have a Strategy Rule base to control this.We also check whether the current state of hockey requires offense or defense before developing a corresponding strategy.Strategy Rule base algorithm can be shown:

Algorithm 2: Strategy Rule base

```
Set default strategy is 3;
if hockey direction is moving away from the robot then
    No need to plan;
if hockey direction is static and in robot half then
    change plan to strategy 2;
else
    set plan to strategy 3;
if plan is strategy 3 and hockey prediction path
  changed then
    if hockey speed is too fast then
        change plan to strategy 1;
else
    execute strategy 3;
```

IV. EXPERIMENT AND RESULT

Our performance test is divided into three parts: system performance, recognition performance and path prediction and planning performance.

System performance corresponds to whether our digital modeling and simulation are stable, and whether its physical effects and confrontation capabilities meet our expectations.

The recognition performance corresponds to our gesture control module. We will test whether the module can correctly recognize user gestures, and test whether the controller responds in a timely manner during actual gameplay.

Path Prediction and Planning is a test for our robot. We will test whether the robot can correctly judge the user's hitting direction, whether it can make a correct decision, and whether the decision can be executed in time.

A. System performance

Because robots come in a large variety of types and configurations, standardisation of robot specifications is difficult [14]. In order to get a suitable and accuracy evaluation on how this robot meet our task, we demonstrate 2 steps for testing robotic system performance.

One is the unit evaluate of the robot, and in this case, we tested the transmission response within different frequency request. The result is shown in Fig 11. In this evaluation, we send same signal(target position curve) with different time width to robot, and then plot the response signal(joint position curve) from the state sensor as well as the physical position(actual position curve). In case the experiment was done in simulation environment, the joint position is almost same as physical position.

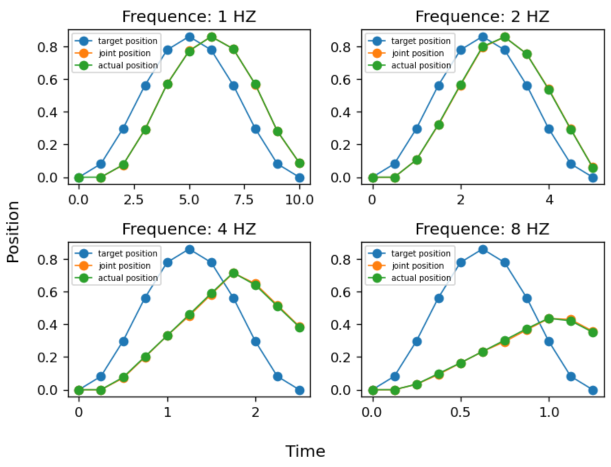

Fig. 11. Track response evaluation.From low frequency to high frequency, the performance is getting worse.Considering trajectory prediction applied to the final system, the result under 4Hz is acceptable.

On the other hand, regrading on overall system, we conducted a 300-minute comprehensive test to evaluate the system's response time to the shot and various possible misoperations. During this process, our system did not crash. When testing a hockey ball with too much force, there is a certain chance that the hockey ball will fly off the table. In this case, we only need to reset hockey ball pose in simulation environment, and our system can still operate normally without downtime.

B. Recognition performance

We train the gesture recognition model using a mini-dataset [4] on 6G RAM GPU(GTX 1660ti).The batchsize is 128, and the last epoch is 150. In training progress, we apply a softmax function on the dense layer output and calculate the categorical cross entropy loss as equation 4 between output and label, then Adam was applied as optimizer for backpropagation.

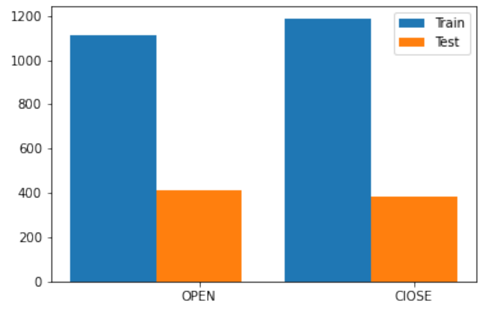

Fig. 12. Dataset summary. It contains totally 3096 images for the 2 types of hand gesture

$$CE = \sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{4}$$

M is number of classes; y is binary indicator (0 or 1) if class label c is the correct classification for observation o; p is predicted probability.

The overall accuracy on test dataset is 0.9766 corresponding to curves in Fig 13 and 14 shown. And the ROC curve is plotted in Fig 15.

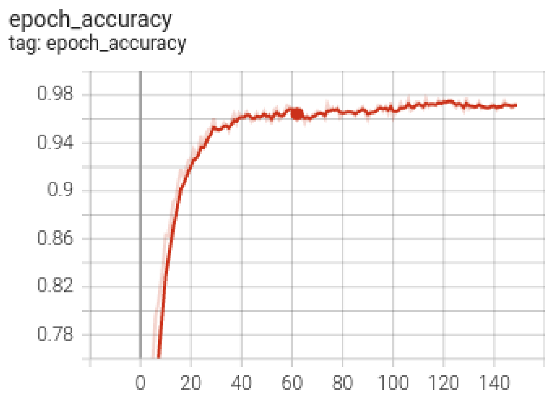

Fig. 13. Validation accuracy

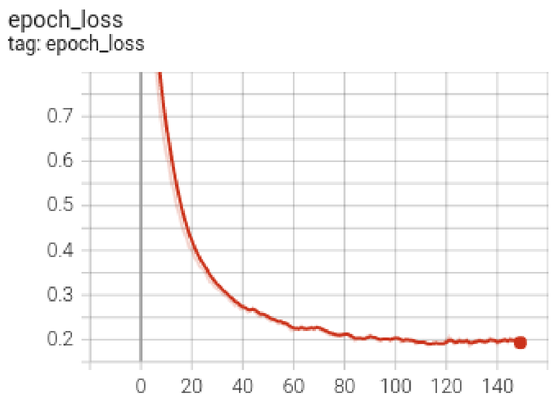

Fig. 14. Validation loss

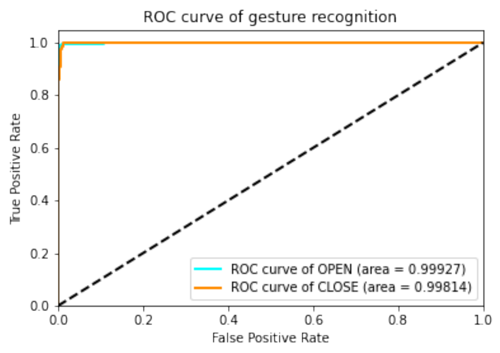

Fig. 15. ROC curve

From the statistical results, we can get the conclusion that this model has excellent performance on specific dataset. However, *how people perceive and interact with a visualization tool can strongly influence their understanding of the data as well as the system's usefulness* [11].In order to better evaluate the actual performance of this model, the intuition of human should be also considered to be an important indicator of model performance. In this case, more than ten volunteers have been invited to participate a test. Since we built a simple website, they can use their webcam in their own home to do it online. They rated our gesture recognition on a 1-10 scale (10 being the best) on accuracy, responsiveness, comfort and learning difficulty: table VI.

Based on the average feedback from testers, we believe that our gesture recognition and control part is adequate for human players and very easy to learn for most people.

TABLE VI
RATING FROM TESTER(1-10 SCALE, 10 BEING BEST)

tester id	accuracy	responsiveness	comfort	learning difficulty 10:easiest to learn
1	9	8	9	10
2	8	7	10	7
3	9	6	8	9
4	7	6	9	8
5	8	7	9	9
6	6	7	8	5
7	8	6	7	4
8	10	9	9	9
9	8	6	9	9
10	9	7	10	9
average	8	7	9	8

C. Path Prediction and Planing

We set 10 times per second to use Planner to decide Strategy.The reaction speed can be handled accurately in normal hitting situations.But if the operator hits the hockey too fast (20m/1s) , the model will occasionally not be able to keep up, resulting in a lag.

In theory, the more frequent planner calls we set, the faster the response will be. However, when we set the call to exceed 50 times per second, due to the asynchronous time, the planner will be called again before a call has ended, resulting in the robot shaking.

Nonetheless, at 10 times planning per second, our robot can still respond in near real-time to the vast majority of shots. This frequency is actually close to the response speed of most ordinary people. In the process of recruiting many friends to participate in the test, they were satisfied with the response speed of the robot.

V. CONCLUSION

Our modeling efforts can correctly represent hockey collisions and scoring goals. The gesture recognition system can recognize gesture controls from the user. The air hockey robot has played against more than a dozen players and achieved multiple victories while using Pressing Attack strategies. In chats with several friends who were invited to test, they expressed their appreciation for the immediacy, stability and comfort of our system. For our counter bots, some of them with experience playing air hockey found our Pressing Attack strategy to be moderately difficult. And for newbies, they think our Defence strategy is too strong.

On our personal computer, our entire system can run at nearly 30 fps with extremely low latency for user gesture control and near-instant robot response. In the confrontation with non-professional air hockey players, our robot can get close to 60% winning rate, and in the confrontation with professional air hockey players, it can also persist in hitting more than 3 rounds in each inning. These met or exceeded our initial expectations.

VI. FUTURE WORK

Our current work only completes the digital modeling part. In the foreseeable future, our project can be further expanded in three main directions:

1. Convert the modeling into the actual physical structure, which users can actually experience without a computer. Through correction and adjustment, it is expected to provide services not only for patients, but also for ordinary air hockey fans. There are already similar products and projects on the market today, and our strengths lie in faster response times and gesture control. These advantages can be obtained through extremely inexpensive improvements on the hardware basis of other products. In the future we could work with the owners of these products to launch enhanced versions of the air hockey robots.

2. Enhance our strategy. Due to time and hardware constraints, we did not apply an overly complex model for policy formulation. However, it is possible to add one or more models for policy formulation. More complex and aggressive strategies can greatly increase the fun and confrontation of air hockey, which are potential benefits of the program.In fact, during development, we saw quite a few air hockey games based on 2D planes rather than simulations, which also designed their own bots and confrontation strategies. Some of these strategies can also take effect in 3D physics simulation. In the future, we can provide a full range of upgrades for our robots by cooperating with these game makers and integrating their strategies. Also, inspiring by *openai_ros* [12], we developed an *Open AI Gym* [13] interface for ROS. This interface was built based on ros-bridge, which can be decoupled from

ROS and can be implement with many existing reinforcement learning algorithms without convert it into ROS environment.

3. By optimizing our code structure, deploy a complete set of online services to provide online services for those who cannot deploy a set of air hockey equipment or powerful computers in their own homes. This requires more complex network construction work, but it can help more people and bring health and happiness to them.When we invited our friends to test, we made a simple web page that allowed them to play remotely using their webcam. This is actually the prototype of cloud gaming. However, there are still quite a few bottlenecks in cloud gaming technology, which all depend on industry progress and technological development. It is expected that we can also expand this project into a kind of cloud game in the near future.

References

[1] Wang, W., Chen, Y., Li, R. (2019). Learning and comfort in Human–Robot interaction: A review. Applied Sciences, 9(23), 5152. doi:http://dx.doi.org/10.3390/app9235152

[2] Russell Smith (2006). Open Dynamics Engine v0.5 User Guide http://ode.org/ode-latest-userguide.html

[3] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg and Matthias Grundmann. MediaPipe: A Framework for Building Perception Pipelines, 2019; arXiv:1906.08172.

[4] Kazuhito Takahashi, hand-gesture-recognition-using-mediapipe https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe.git

[5] Siddiqui, M.A., Qadri, M.T., Siddiqui, A.A. et al. Autonomous Air Hockey Robot. Wireless Pers Commun 95, 641–653 (2017). https://doi-org.libproxy1.nus.edu.sg/10.1007/s11277-016-3916-2

[6] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[7] Chang CL, Chen ST, Chang CY, Jhou YC. Application of Machine Learning in Air Hockey Interactive Control System. Sensors (Basel). 2020;20(24):7233. Published 2020 Dec 17. doi:10.3390/s20247233

[8] https://github.com/JJulio/AHRobot

[9] Weidman, P., Sprague, M. (2015). Steady and unsteady modelling of the float height of a rotating air hockey disk. Journal of Fluid Mechanics, 778, 39-59. doi:10.1017/jfm.2015.374

[10] Mark W. Spong,Impact controllability of an air hockey puck,Systems Control Letters,Volume 42,Issue 5,2001,Pages 333-345,ISSN 0167-6911,https://doi.org/10.1016/S0167-6911(00)00105-5.

[11] M. Tory and T. Moller, ”Human factors in visualization research,” in IEEE Transactions on Visualization and Computer Graphics, vol. 10, no. 1, pp. 72-84, Jan.-Feb. 2004, doi: 10.1109/TVCG.2004.1260759.

[12] Alberto Ezquerro, Miguel Angel Rodriguez and Ricardo Tellez, OpenAI ROS http://wiki.ros.org/openai_ros

[13] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang and Wojciech Zaremba.

[14] H. Van Brussel,Evaluation and Testing of Robots,CIRP Annals,Volume 39, Issue 2,1990,Pages 657-664,ISSN 0007-8506, https://doi.org/10.1016/S0007-8506(07)63002-9.
OpenAI Gym, 2016; arXiv:1606.01540.