EE267 Project Proposal
Yang Li, Huizi Mao

Problem Statement:

What we are trying to do is to combine a VR headset and a depth camera to make a AR headset. The depth camera captures the surroundings with depth info and pass it through the computer onto the display. A virtual object should be rendered on the display at the same time. To make it easier, we'd like to first render a tennis ball, that the user can toss it and the tennis ball should be able to bounce around. If that can be easily implemented, we'd like to design a dog that can run around in the sce
ne chasing the tennis ball.

What's out there

A couple teams have already tried similar projects. For instance, a team from TNG Tech created a Terminator style AR view. What they did was to assemble the Rift SDK with RealSense SDK using C#. They also brought in some advanced features such as facial tracking and recognition, speech recognition, heart rate detection, etc. A team from KomuroLab did a see-through AR system as well[1]. They also used a depth camera and projected the real scene captured together with virtual objects onto the display. Their project was not exactly VR, but the thing we can learn from their project is collision detection using depth camera[2].

What's new

Basically all current see-through AR based on VR and depth camera is an assembling of the two devices. But what's new about our project is, we'd like to combine the see-through VR with physics engine and collision detection based on depth info. Specifically, when the user tosses the tennis ball, the ball should be able to move based on real physics rules - gravity and momentum. Also, collision detection will be introduced so that the ball is a able to bounce when hitting any object in the real world.

Timeline:

1. 5/30 - The first job is to get the depth camera hooked up with the headset and display the surroundings in the headset realtime. Specifically, we should try to decrease the delay. If the delay is really bad, another option we can try is to use a panorama camera and simulate the rotation of head with IMU.
2. 5/31 - Next job is to render a virtual tennis ball, and display it on the headset. This ball should be able to move around and rotate(optional as it's a ball) when given a force. The background can just be all black for now.
3. 6/2 - Next step is to integrate the virtual ball into the captured surrounding. Specifically when user's hand get in the view, there should be a ball in hand. When user drops or tosses the ball, the ball should be released with an initial velocity. If this is too hard, an alternate can be speech controlling, i.e. user says a command such as "drop" the ball, and then the ball should be dropped from above.

4. 6/5 - This is the most important thing - collision detection should be implemented. We should know when the ball hits anything, and what the physics are. This is like the physics engine in Unity. But the hard thing is we are dealing virtual and real objects at the same time.
5. 6/7 - The last step is to implement consistent occlusion between real and virtual objects, for example when the ball rolls to the back of a table, it should be invisible.
6. (Optional) If we got enough time left, we will create another virtual object - a dog. Ideally we should use some pre-designed dog model. And the virtual dog should have consistent occlusion with real objects. The virtual dog should avoid hitting any real object.
7. 6/9 - Final Project Poster Session

References

[1] Yuko Unuma and Takashi Komuro: Natural 3D Interaction using a See-through Mobile AR System, Proc. The 14th IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2015), pp. 84-87 (2015).
[2] Yuko Unuma and Takashi Komuro: 3D Interaction with Virtual Objects in a Preciselyaligned View using a See-through Mobile AR System. ITE Trans. on MTA Vol. 5, No. 2, pp. 49-56 (2017)