Oculus Rift Application for Training Drone Pilots

Evan O'Keefe
School of Computer Science University College Dublin Ireland; evanokeefe@gmail.com

Abraham Campbell
School of Computer Science University College Dublin Ireland; abey.campbell@ucd.ie

David Swords
School of Computer Science University College Dublin Ireland; david.swords@ucd.ie

Debra F. Laefer
School of Civil Engineering University College Dublin Ireland; debra.laefer@ucd.ie

Eleni Mangina
School of Computer Science University College Dublin Ireland; eleni.mangina@ucd.ie

ABSTRACT

The research described in this paper, focuses on a virtual reality headset system that integrates the Oculus Rift VR headset with a low cost Unmanned Aerial Vehicle (UAV) to allow for drone teleoperation and telepresence using the Robot Operating System (ROS). We developed a system that allows the pilot to fly an AR Drone through natural head movements translated to a set of flight commands. The system is designed to be easy to use for the purposes of training drone pilots. The user simply has to move their head and these movements are translated to the quadrotor which then turns in that direction. Altitude control is implemented using a Wii Nunchuck joystick for altitude adjustment. The users use the Oculus Rift headset a 2D video stream from the AR Drone, which is then turned into a 3D image stream and presented to them on the headset.

CCS CONCEPTS
Software engineering → Distributed Computing;

KEYWORDS
Robotics, Robot Operating System, Tele-operation, Unmanned Aerial Vehicle, human computer interaction

1 INTRODUCTION

With the advent of the Oculus Rift virtual reality headset, virtual reality headsets have reduced in prices that are now in reach of the consumer market. Teleoperation in robotics with head tracking head mounted devices is becoming a more popular research field due the cost of this technology and also the interest in naturally controlling the movement of robots using our own movements, versus that of a hand based controller such as a joystick or game pad. Such operations allow users to use robots with incredible ease as they are formed around our own natural kinematics. This in practical terms means users theoretically require little to no training to become competent users of these systems, allowing more time to be spent on the task at hand rather than waiting for the user to learn how to use the system before being able to solve the problem. The main problems with virtual reality technology are the difficulty of emulating natural movements and the required training time. It has been noted that the training time can vary based on the complexity of the system and the main user interaction feedback loop, through the utilisation of different controllers. Conventional control systems do not provide feedback on time for optimal control and it is the same for the area of aerial robotics.

In the last decade, the technological improvements and the cost reduction of Unmanned Aerial Vehicles (UAVs) have increased their applications. Nevertheless, the drone pilots still have to be trained through pre-defined hours of practice, with real time view of the drone as a requirement. Furthermore, today's controllers provide no feedback as to the drone's state and the pilot must have visual contact during the flight. As a consequence, it prevents effective UAV deployment in scenarios such as search and rescue, where an operator needs situational awareness for safe navigation.

Within this paper we present the integration of the Oculus Rift into a modular framework called the Robot Operating System (ROS)[1] and test it by controlling a quadcopter using mainly the gyroscope and accelerometers in the headset. By utilising ROS to create our application we anticipate other users to be able to easily recreate this setup provided they have the relevant hardware to use it.

2 BACKGROUND RESEARCH

Although Virtual Reality (VR) came to public's attention in the 1980s and 1990s, it is only the last decade that the headsets have seen such an increase in sales. Several applications within gaming industry and education have moved the VR to be used outside simulations only for the military use. David Gossow at Willow Garage [2] integrated the Oculus Rift virtual reality headset into RViz and created a package for the PR2 robot called PR2 Surrogate. It lets the user teleoperate a PR2 using the Oculus Rift and the Razer Hydra game controllers. Using the Oculus Rift, turning the user's head is mirrored by the PR2 surrogate allowing the teleoperator to feel themselves are present in their environment and allows the user to gain a better spatial grounding of their surroundings using the headset. It allows the user to operate the robotic hands with the Razer Hydra with greater precision as the humans operators are more used to interacting with objects with first person spatial data than by traditional 2D screen displays. Pfeil et al [3] present a study exploring upper body 3D spatial interaction metaphors for control and communication with Unmanned Aerial Vehicles (UAV) such as the Parrot AR Drone. The focus of this research was to interpret users spatial proximity to an Xbox Kinect 3D camera into movements which are then sent to a drone. But in this case the perception from the user is personal and do not have the view from the drone's position.

Pittman et al [4] from the University of Central Florida have implemented a similar system to one we intend to implement by integrating the equipment into a single standalone application. Higuchi et al [5] have developed a head tracking solution for the teleoperation of a Parrot AR Drone using synchronized optical trackers attached to the head and drone chassis called Flying Head. This essentially allows the users height, limb movements and head actions to be mimicked by the Ar Drone. Mollet and Chellali [6] from the Italian Institute of Technology developed a head-tracking system, combined with a VR helmet, which allows the teleoperators to see in a natural way what the robot see. Additionally, it allows us to add some Augmented Reality features, like for example virtual arrows to represent points of interest like another robot, an identified object to manipulate, maps, unknown areas, etc. Robonaut [7], a humanoid designed for use in space, was designed with a teleoperation technique for control of the entire robot, including the head with two degrees of freedom. Users wore an HMD with head tracking and were able to send commands to the Robonaut by rotating head yaw and pitch. This control scheme is comparatively simple, as it only tracks two axes of rotation of the head, while our techniques makes use of three axes of head rotation and three translation axes. Possible applications of this technology and methodology spans to areas such as surveillance [8] , search and rescue [9] and robot surrogacy [10], exploration of unsafe environments such as unstable nuclear power plants [11] and mining [12] as examples of where they would be significant improvements over traditional control systems.

The goal of this project is the development of the software and/or introduction modifications for the control of a Parrot AR.Drone 2.0 quad-rotor using an Oculus Rift virtual reality headset. The Parrot AR.Drone 2.0 is a standard configuration quad-rotor helicopter with a nylon and carbon fiber construction measuring 57cm across. It is capable of both indoor and outdoor deployment. The pitch, roll and yaw of the quad-rotor is detected using an inertial measurement unit, 3 axis gyroscope, accelerometer and magnetometer. Altitude is measured by an ultrasonic altimeter, with a range of 6m, with the addition of an air pressure sensor. The quad-rotor is powered by a 11.1V lithium polymer battery, providing approximately 12 minutes of flight time. There are two built in cameras, one forward facing, the other downward facing. Wireless interface is 802.11n and also can be manual connected via USB. The Oculus Rift [13] is a virtual reality head mounted display. The headset fully obscures the wearers view of the real world, allowing for an immersive experience. The internal display is 1280 x 800 with a 90 degree field of view. In addition to this, the headset includes 3 axis gyros, accelerometers and magnetometers at 250Hz, allowing for tracking with little latency. The headset is not wireless, and remains connected to a control box. The control box has DVI and HDMI interfaces. A USB interface is provided to gain access to tracking data via a host ma- chine. With the Oculus Rift (OR), only the on-board sensors are used to track the user's movements by fusing the sensor readings and efficient movement tracking. The approach followed within this project has shown that the virtual reality headsets can play a vital role in telepresence and teleoperated UAVs.

3 IMPLEMENTATION

3.1 OculusRotor

The OculusRotor system is based on the synergy between the ROS architecture and the AR Drone SDK's architecture for Linux. The AR Drone Autonomy ROS library handles separate threads associated with the ROS node applications. The applications run o separate threads like the AT Commands (representing the basic functionality); the Navidation data (containing the AR Drone in- formation); the Video management; the Video recorder (on the device); the Control thread requests and the AR Drone Acedemy (for UAV image capture and rmware). All of this functionality is managed by the ROS Node provided by Simon Fraser University and maintained by Mani Monajjemi other contributors [14].

3.2 Interpreting Head Gestures

The ROS Twist vector messages are utilised for the AR Drone to be own in order for the UAV to remain within WiFi access point. The Oculus Rift headset would receive these messages and store them as baseline values. The changes would then be interpreted into a 6 degree of freedom twist message and would be delivered as a message to the AR Drone. The Oculus ROS node publishes Quaternion [15] messages which are simply four values x, y, z, w, which represent rotations about each of the x, y, z planes. These messages are received and translated to Euler angles. Quaternions can be used in avionics but seemed more advanced than what was needed for this project and would require more time to implement in the system. The Euler angles also make more sense for the quad copter as Gimbal lock only becomes a problem when the yaw of the drone would reach the peak i.e the front of the drone is pointing directly upwards, causes a 1 degree of loss as the yaw and pitch gimbals lie on the same plane. For this project this issue was not a problem, as if the drone ever reached this point, it would mean the drone is about to flip over to stabilise itself, possibly causing the user some disorientation or possibly leading the drone to a collision with either and object or the ground.

3.3 Experiment Setup

The experiment was set up with twenty users, half of them experienced and half novice. Every participant had to test the system in two separate operations, one flight utilising the stick control and one flight the headset. The area for testing was the ground floor of O’Briens building at University College Dublin, as shown in Figure 1. The users had to take off, follow a specific path within a distance of thirty meters, make a turn and land at the point of take off.

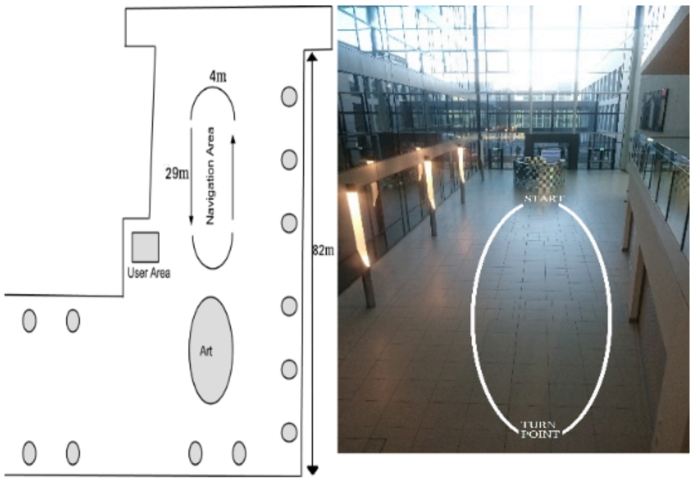

Figure 1: Flight map and area

3.3.1 ROS Message Structure. To prevent messages over owing in the system certain messages interrupt the main loop creating a precedence for them. Examples of these are messages to set drone state to takeoff, land, reset or emergency land. When these are activated a timed loop is activated that makes sure these messages are the only ones being sent and prevent other messages from being sent. The loop timers are long enough for the drone to land even at maximum height from the ground. There exist two twist messages in the node. One for the linear and angular translations and another altitude adjustments. The altitude receives lower priority as the drone was prone to start drifting at times. This allows the user to avoid collision by correcting for this drift.

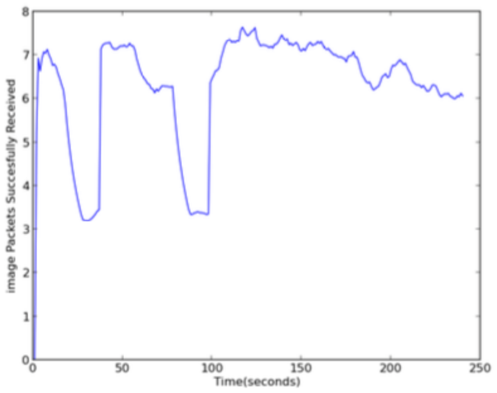

Figure 2: Drone image transmission

3.3.2 Monocular First Person Viewer. The viewer worked by taking the single 2D image and performing left and right eye projections on the image then putting the images through OpenGL barrel distortion and chromatic aboration shaders, which are provided in the Oculus Rift SDK . This produces the 3D images. OpenCV2 was used to overlay images onto the video stream before the transformations to provide basic information to the user. Messages were sent to the Oculus Rift headset directly providing battery, flight state and altitude to the user, to allow them to use the drone knowing how much power it had. This allows the user to know how far and high away they can use the drone, with certain battery levels to avoid losing visual feedback and command latency.

3.4 User Flight feedback

The effectiveness of the system was evaluated based on different approaches (packet and image performance; user's interaction with the system during flight operation; user's survey after the flight operation). The Ardrone autonomy ROS node was set to run at a frequency of 1000Hz to allow the user to have most current sensor feedback such as visual image feedback and altitude estimations and make the experience as close to real time as possible through the interpretation of the Oculus Rift sensor data. This was done to lower the latency of head gesture commands to the drone. By lowering the latency in commands we aimed for better perception of movement in the drone, which would also allow the user to avoid collisions. The user flight system was then evaluated between novice and experienced users in terms of the control modes and a webcam was used to record the facial expressions of the users, while they were flying the UAV through the Oculus Rift headset.

3.5 Visual command latency

Within this project experimentation, the recording of the packet loss on both the image stream and command streams using the rosbag utility took place. The UAV was controlled through the movements of the Oculus Rift headset. As the AR Drone was own by the user around the testing area, the packet loss of the drones visual feed and command publishing were monitored to make sure the user didn't lose control of the drone and visual feedback by surpassing the WiFi signal limits. Published headset commands were stable over the duration of the tests, while the visual feedback as shown in Figure 2, required more bandwidth and it was unstable during the tests.

4 CONCLUSIONS

The AR drone worked very well in indoor environments with no wind conditions but was never tested outside. It would be interesting to see how well the system works in an open area with wind conditions. The AR Drone has wind speed sensors allowing it to correct for wind up to a point. It would be interesting to see how responsive the system would feel to the user with this outside influence. Using the new Oculus Rift Consumer edition with the user tracking would be interesting to integrate with this project as it might be used to eliminate the use of the wii controller. The drone image quality would be improved with the display resolution so long as the ardrone autonomy library is modified to use the 720p camera resolution. The drone's vision through the Oculus Rift presented the problem that the user losses the depth perception needed to avoid collisions with objects. This was expected and can be avoided after some test flights, by allowing user's eyes to adjust to estimating the distances. We have concluded that this system would require a collision avoidance algorithm that would deal with collision avoidance using monocular vision. The best solution provided seems to be monocular localisation and mapping library called PTAM based on the paper by Klein et al [16]. This would require the drone to have a calibrated camera with known focal length to estimate the pose of the camera and generate a map of the room. Using this map then a system could be built using automatic collision avoidance with teleoperated inputs.

The AR drone responded well to the commands being published to it. The video latency was found to work in ranges of less than 60 meters. Anything over that would reduce the performance of the system. Loss of information and lack of responsiveness hinder the users input as the image feed would be obstructed the further out they got. The user would sometimes need to use the drone compass to guess where to y the drone to bring it back within acceptable distance of the users WiFi. The system was tested with some users who hadn't operated the drone before and they showed they were able to y the system with minimal help. 75% of the user that had prior drone piloting experience were able to utilise the compass, but those that had no prior experience could not navigate back due to the drop out rate of the visual frame.

5 ACKNOWLEDGEMENT

Thanks to the UCD School of Computer Science Informatics for the use of the AR Drone and Oculus Rift equipment and the use of the Robotics lab.

REFERENCES

[1] ROS(2014). Robot Operating System, http://www.ros.org/.

[2] Gossow D., Leeper A., Hershberger D., Ciocarlie M.,(2011), Interactive markers: 3-d user interfaces for ros applications, IEEE Robotics & Automation Magazine, 2011

[3] K. Pfeil, S.L.Koh, and J. La Viola. Exploring 3d gesture metaphors for interaction with unmanned aerial vehicles. In Proceedings of the 2013 International Conference on Intelligent User Interfaces, IUI '13, pages 257-266, New York, NY, USA, 2013. ACM

[4] C. Pittman and J.J. La Viola, Jr. Exploring head tracked head mounted displays for rst person robot teleoperation. In Proceedings of the 19th International Conference on Intelligent User Interfaces, IUI '14, pages 323-328, New York, NY, USA, 2014. ACM

[5] K. Higuchi, K. Fujii, and J. Rekimoto. Flying head: A head-synchronization mechanism for flying telepresence. In Artificial Reality and Telexistence (ICAT), 2013 23rd International Conference on, pages 28-34, Dec 2013

[6] N. Mollet and R. Chellali. Virtual and augmented reality with head-tracking for efficient teleoperation of groups of robots. In Cyberworlds, 2008 International Conference on, pages 102-108, Sept 2008.

[7] W. Bluethmann, R. Ambrose, M. Diftler, S. Askew, E. Huber, M. Goza, F. Rehnmark, C. Lovchik, and D. Magruder. Robonaut: A robot designed to work with humans in space. Autonomous Robots, 14(2-3):179-197, 2003.

[8] F. Vexo, S. Cardin, D. Thalmann and X. Righetti, "Immersive flight for surveillance applications," 2007 IEEE Symposium on 3D User Interfaces(3DUI), Charlotte, NC, 2007, pp. null. doi:10.1109/3DUI.2007.340786

[9] H. Martins and R. Ventura. Immersive 3-d teleoperation of a search and rescue robot using a head-mounted display. In Emerging Technologies Factory Automation, 2009. ETFA 2009. IEEE Conference on, pages 1-8, Sept 2009

[10] Y. Tokuda, A. Hiyama, T. Miura, T. Tanikawa, and M. Hirose. Towards mobile embodied 3d avatar as telepresence vehicle. In C. Stephanidis and M. Antona, editors, Universal Access in Human-Computer Interaction. Applications and Services for Quality of Life, volume 8011 of Lecture Notes in Computer Science, pages 671-680. Springer Berlin Heidelberg, 2013

[11] M. D. McKay and M. O. Anderson. Telepresence for mobile robots in nuclear environments, 1996

[12] D. Hainsworth. Teleoperation user interfaces for mining robotics. Autonomous Robots, 11(1):19-28, 2001

[13] OVR(2014).OculusRift,https://www.oculus.com/.

[14] Mani Monajjemi (AutonomyLab, Simon Fraser University) (2015) http://bebop-autonomy.readthedocs.io/

[15] Quartenion:http://mathworld.wolfram.com/Quaternion.html

[16] G.KleinandD.Murray.Parallel tracking and mapping for small AR workspaces. In Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07), Nara, Japan, November 2007.