Project EPR400/402 **First semester report**	**August** **2022**	**Note:**

Student to complete this section					
Williams	**M.L.**	**Mr.**	**18013555**		
Real-time hand gesture control of a virtual object in augmented reality			**Study leader:** **Mr. H. Grobler**		
Degree programme enrolled for: Computer Engineering	**Project number:**		HG9	**Revision number:**	0
Student declaration: I understand what plagiarism is and that I have to complete my project on my own. I am fully aware of the University's policy in this regard. I have not used another student's past or present written work to submit as my own. I declare that this report is my own original work. Where other people's work has been used (either from a printed source, internet or any other source), this has been properly acknowledged and referenced in accordance with departmental requirements.	Student signature			05 August 2022 Date	

Declaration by language editor	
I have been allowed adequate time to read this document carefully and to make corrections where necessary (date received indicated below). To the best of my knowledge, correct formatting, spelling and grammar are used throughout the document.	
M.L. Williams (language editor)	05 August 2022 Date

Table of Contents

1. Literature study

With the increase in the proliferation of powerful personal computing hardware it has become feasible to create augmented reality applications that integrate virtual objects with a user's physical environment. Similarly, modern computer systems can perform real-time inference on a large range of alternative inputs and return useful results – this has led to the advent of human-control inputs to computers like hand gesture control.

These two sub-fields - augmented reality and gesture control, can be combined to give a user a natural and intuitive control mechanism for interactive and visual applications. The literature is studded with examples of applications that use this combination of technologies, such as Billinghurst [5] who utilizes a Microsoft Kinect depth and RGB camera to treat agoraphobia by creating virtual spiders that the user can interact with using their hands in an augmented reality application. Baldauf [3] uses gesture input from a mobile phone camera to select, shrink and zoom in on virtual objects presented in the environment as well as to recognize gesture volume controls for a music application.

The ability to locate virtual objects in the context of the real-world environment in an augmented reality application is important if realistic interaction is to take place. Kato [1] implements a tabletop augmented reality application for handling small virtual shapes that relies upon a global coordinate system and paper tracking fiducials placed on the tabletop to give both virtual objects and real-world objects their coordinates in the global coordinate system and then be able to control virtual object movement and behavior accordingly. Similarly, Buchmann [2] uses a world coordinate system in an urban planning augmented reality application that tracks the position of virtual objects as well as the user's hand and current gesture to determine if an object should be grasped, moved or released at any given time. This also allows for collision avoidance as the same coordinate system is shared by all objects – real or virtual.

These applications receive hand gestures as input and hand gesture control itself can be considered as the two sequential problems of hand pose estimation and gesture recognition based on the hand pose predicted. Gesture recognition is most often performed using machine learning approaches such as support vector machines, Naïve-Bayes classifiers and convolutional neural networks as by Ahmed [11] for the recognition of Indian sign language based on hand coordinate input. It can also be accomplished by extracting features from the input image using Gabor Wavelet Transforms and gradient local-auto correlation and then providing these features to a multi-layer perceptron or K-nearest neighbors system such as by Sadeddine [15] to recognize sign language. The complexity of the algorithm required in gesture recognition depends on the static or dynamic nature as well as diversity of the input gestures.

What is apparent from the literature, however, is that the main challenge of gesture recognition is first acquiring an estimation of the user's hand pose from camera input – solutions to this problem have been proposed and implemented since the 1990s. These early solutions [4] relied on classical approaches to hand pose estimation such as using The Continuously Adaptive Mean Shift algorithm to recognize very high or low saturation of image pixels to segment a hand from its background and then using a curvature-based least-square fitting algorithm for detecting the contours of the hand such as fingertips. An alternative to hand pose estimation is to use a physical glove with fiducial markers on it as used by Buchmann [2] to detect the position of a user's hand in space. However, with the advent of modern computing power and the rise of deep learning, the literature has been saturated with machine learning approaches to hand pose estimation that require none of the special hardware or highly specific algorithms that previous implementations required.

A state-of-the-art hand-tracking application created by Google - dubbed Mediapipe Hands [13], uses a series of convolutional neural networks to train a palm detector and hand landmark model to output coordinates of hand landmarks. The system runs in real-time on mobile devices and is trained using real images of hands as well as synthetic hand models. Similarly, Qing [9] uses a deep convolutional neural network with just convolutional and pooling layers to output three-dimensional joint locations for a hand based on webcam input. This is why the literature often refers to hand pose estimation as hand joint-regression. Gomez-Donoso [10] employs a similar architecture to predict joint locations by first using a convolutional neural network to detect and segment the hand using a box prediction system reminiscent of the YOLO9000 architecture [8], and then regress the joints of the hand using a large convolutional neural network based on the RESNET50 architecture [7].

Alternatively, there are implementations of hand pose estimation that also make use of depth camera input. This depth input is often modified in an intermediate transformation such as a heatmap to show where each joint of the hand is likely to be and regresses the location of the joints from this intermediate layer. This is the approach taken by Chen [12] where a convolutional neural network regresses joint locations from feature regions which are themselves extracted from feature heatmaps created by depth image input. Ding [14] and Ge [6] also make use of heatmaps of joint coordinates and subsequent fine-tuning algorithms to output joint locations based on the intermediate layers.

It is evident that the advent of deep learning has yielded a large number of machine learning approaches to hand pose estimation and that the extensive use of convolutional neural networks is the modern approach most preferred in academia. This is due to the ease of not having to implement detailed representations of low-level hand shapes, patterns and methods of identifying these features in input imagery but rather instead training a machine learning system to identify and learn these low-level abstractions using vast amounts of training data and machine learning architectures such as the convolutional neural network.

In conclusion, to implement a system that can perform hand gesture control of a virtual object in augmented reality it is evident that the preferred approach in the literature for such a task is to use a deep-learning architecture to regress hand joint coordinates from either a depth or standard RGB camera input. Gesture recognition can either be performed by machine learning approaches or by classical calculations depending on the complexity of the required gestures. Augmented reality and the combination of virtual reality objects with real-world objects can create immersive and useful applications when a suitable input camera is used and a shared coordinate system is established to track both virtual and real objects and prevent collisions between them. A system will thus be developed that can accept user gesture input using a deep-learning approach and then translate that gesture into meaningful instructions for a virtual object present in an augmented reality scene that presents realistic interactions between the objects and uses a global coordinate system to prevent object collisions.

2. Work breakdown and first semester progress

The work breakdown for the entire project is shown in Table 1. Progress indicated is up to 25 July.

Task	Progress	Brief description
Background reading and research	100% complete	A number of gesture control as well as augmented reality research papers were read and saved for later use in the literature review.
Conceptual design of whole system	100% complete	The overall design of the system (neural network gesture classifier and virtual object rendering pipeline) was decided upon based on the research papers read and constraints enforced by the project budget and module requirements.
Preparation of project proposal	100% complete	Rev 1 was approved.
Hand gesture prototype	100% complete	Using the Mediapipe library, a prototype was constructed that can recognize nine distinct gestures and track a user's hand from webcam input.
Virtual object prototype	100% complete	Using the OpenGL library, a prototype was constructed that renders a 3D cube on top of a video input and can move the cube around with user keyboard input.
Integrated prototype	100% complete	The two prototypes above were integrated so that a user could move their hand to control a virtual cube superimposed on the webcam input.

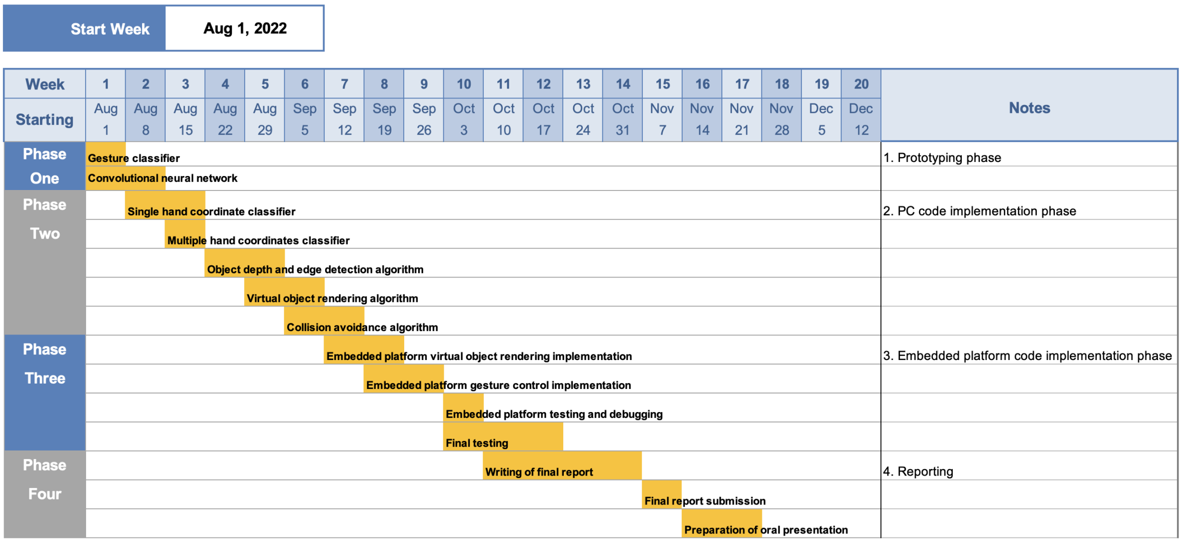
Depth sensor interfacing and installation	100% complete	The virtual object requires depth data of the environment it is to be rendered into and so a Microsoft Kinect sensor was sourced and the necessary libraries installed in order to begin prototyping with it.
Fully-connected neural network	100% complete	A basic feed-forward neural network was implemented in Python and demonstrated learning on a toy dataset.
Convolutional neural network	85% complete	All components of the CNN were implemented but testing, debugging and improving training time with mathematical optimization is still ongoing.
Single hand coordinate classifier	40% complete	A small dataset was built up of images of single hand digits and testing begun but the training loss is still very high and unacceptably slow – refinement to the architecture and underlying algorithms is needed.
Multiple hand coordinate classifier	5% complete	Work on this task is pending completion of the single hand coordinate classifier. A dataset of multiple hand coordinates has begun to be compiled.
Gesture classifier	85% complete	The fully-connected neural network prototype was modified to accept hand coordinates as input and output a predicted gesture. Consistently accurate predictions were made with live input data but an increase in accuracy can be achieved by modifying the training dataset of images to include more gestures in different orientations.

Object depth and edge detection algorithm	10% complete	Earlier convolution filter prototypes were modified to detect edges in the input video from the Kinect sensor and an outline of the algorithm needed to combine this data with the actual depth values of each pixel from the Kinect sensor was created.
Collision avoidance algorithm	5% complete	A list of requirements that the collision avoidance algorithm must fulfil was written and the inputs and outputs of the algorithm were defined (depth and spatial data is received while virtual object position instructions are outputted).
Virtual object rendering algorithm	30% complete	Extensive prototyping with the low-level graphics library OpenGL was performed and work begun on how to convert the collision avoidance algorithm's results into OpenGL rendering instructions in the form of an API.
Embedded platform virtual object rendering implementation	0% complete	
Embedded platform gesture control implementation	0% complete	
Embedded platform testing and debugging	0% complete	
Final testing	0% complete	
Writing of final report	0% complete	
Preparation of oral presentation and demonstration	2% complete	Slides have been created for two group presentations given to the research group and a design framework for the presentation decided upon.

Table 1: Work breakdown and progress

3. Project plan (second semester)

The proposed project plan for the second semester outlining the remaining work to be done and estimated completion dates is presented in Figure 1.



Start Week	Aug 1, 2022

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Notes
Starting	Aug 1	Aug 8	Aug 15	Aug 22	Aug 29	Sep 5	Sep 12	Sep 19	Sep 26	Oct 3	Oct 10	Oct 17	Oct 24	Oct 31	Nov 7	Nov 14	Nov 21	Nov 28	Dec 5	Dec 12	
Phase One	Gesture classifier																				1. Prototyping phase
	Convolutional neural network	▨																			
Phase Two		Single hand coordinate classifier	▨																		2. PC code implementation phase
			Multiple hand coordinates classifier																		
				Object depth and edge detection algorithm	▨																
					Virtual object rendering algorithm	▨															
						Collision avoidance algorithm	▨														
Phase Three							Embedded platform virtual object rendering implementation	▨													3. Embedded platform code implementation phase
								Embedded platform gesture control implementation	▨												
										Embedded platform testing and debugging											
										Final testing	▨	▨									
Phase Four											Writing of final report	▨	▨	▨							4. Reporting
															Final report submission						
																Preparation of oral presentation	▨				

Figure 1: Project plan for the second semester

4. References

[1] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, “Virtual object manipulation on a table-top ar environment,” in *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, Ieee, 2000, pp. 111–119.

[2] V. Buchmann, S. Violich, M. Billinghurst, and A. Cockburn, “Fingartips: Gesture based direct manipulation in augmented reality,” in *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, 2004, pp. 212–221.

[3] M. Baldauf, S. Zambanini, P. Fröhlich, and P. Reichl, “Markerless visual fingertip detection for natural mobile device interaction,” in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, 2011, pp. 539–544.

[4] Y. Shen, S.-K. Ong, and A. Y. Nee, “Vision-based hand interaction in augmented reality environment,” *Intl. Journal of Human–Computer Interaction*, vol. 27, no. 6, pp. 523–544, 2011.

[5] M. Billinghurst, T. Piumsomboon, and H. Bai, “Hands in space: Gesture interaction with augmented-reality interfaces,” *IEEE Computer Graphics and Applications*, vol. 34, no. 1, pp. 77–80, 2014.

[6] L. Ge, H. Liang, J. Yuan, and D. Thalmann, “Robust 3d hand pose estimation in single depth images: From single-view cnn to multi-view cnns,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3593–3601.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[9] Q. Fan, X. Shen, Y. Hu, and C. Yu, “Simple very deep convolutional network for robust hand pose regression from a single depth image,” *Pattern Recognition Letters*, vol. 119, pp. 205–213, 2019, Deep Learning for Pattern Recognition, ISSN: 0167-8655.

[10] F. Gomez-Donoso, S. Orts-Escolano, and M. Cazorla, “Accurate and efficient 3d hand pose regression for robot hand teleoperation using a monocular rgb camera,” *Expert Systems with Applications*, vol. 136, pp. 327–337, 2019, ISSN: 0957-4174.

[11] H. F. T. Ahmed, H. Ahmad, K. Narasingamurthi, H. Harkat, and S. K. Phang, “Df-wislr: Device-free wi-fi-based sign language recognition,” *Pervasive and Mobile Computing*, vol. 69, p. 101 289, 2020, ISSN: 1574-1192.

[12] X. Chen, G. Wang, H. Guo, and C. Zhang, “Pose guided structured region ensemble network for cascaded hand pose estimation,” *Neurocomputing*, vol. 395, pp. 138–149, 2020, ISSN: 0925-2312.

[13] F. Zhang, V. Bazarevsky, A. Vakunov, *et al.*, *Mediapipe hands: On-device real-time hand tracking*, 2020.

[14] L. Ding, Y. Wang, R. Laganière, D. Huang, and S. Fu, “A cnn model for real time hand pose estimation,” *Journal of Visual Communication and Image Representation*, vol. 79, p. 103 200, 2021, ISSN: 1047-3203.

[15] K. Sadeddine, F. Z. Chelali, R. Djeradi, A. Djeradi, and S. Benabderrahmane, “Recognition of user-dependent and independent static hand gestures: Application to sign language,” *Journal of Visual Communication and Image Representation*, vol. 79, p. 103 193, 2021, ISSN: 1047-3203.