Frans Ojala 013865821 Monday 25 April 2016

Detecting mid-air gestures done with haptic glove

Interactive Systems

Introduction

Mid-air gestures will likely be at the forefront in controlling augmented and virtual reality scenarios. There is abundant research on gesture recognition on 2D and 3D surfaces but still there remains a gap between the user performing gestures mid-air and the device. We present a prototype that uses a simple, trainable and quite reliable 1$ gesture recognition algorithm to detect gestures performed mid-air. Our prototype does not use cameras, nor complex signal preprocessing thereof, but an orientation sensor embedded within a glove to track hand movement. Further, we present the results of a user study that aimed to test the usefulness of the proposition. Preliminary results show that the activation of the gesture input plays a crucial role in recognising gestures as well as user satisfaction. Gesture recognition of the type proposed in this note may be useful in further prototypes or even commercial products, such as Google Streetview as a lightweight recogniser in the web application should it be used in an augmented reality setting.

Need finding and ideation

Prior to starting work on any prototypes some preliminary user studies were performed. This was performed as intercept interviews. The purpose of the interviews was to find out what people thought of the upcoming paradigm of Virtual and Augmented Reality (VR, AR). The questions included:

1. What augmented reality devices are they aware of and have they used them?
2. Describe the first thing that comes to mind that you would to with AR?
3. If you had an AR and Google streetview, how would you control movement in it?

Both men from the intercept interviews were in their mid to late 20's. Neither had used modern AR or VR technology before, but one had used a primitive form of AR. Both responded that they had not used the technology before because of scarcity and price. One of the participants continued that in his opinion the current state-of-the-art is too big and clumsy to benefit real life scenarios. Referring to the second question, one responded they would play games, and the other responded they would mainly like a heads-up-display with various information. Both respondents gave examples of various gestures, voice control, head and eye tracking as examples of controlling the perceived AR / VR environment.

During a short ideation session we came up with the following use cases, particularly for augmented reality devices:

- sign language translation, to bridge the gap between the deaf, speech impaired and normally hearing and speaking people
- immersive content, including computer augmented reality games
- thumbs-up glove, for easily liking and/or sharing in social media
- contact-lens with embedded display and/or computing system, to bring all functionality to the eye and no glasses needed - minimum disruption
- spinning-app and generator, which would generate electricity for the device and others, while providing an artificial scenery instead of the gym

There are a plethora of use cases in addition to these, that may have untapped potential to create value in life. However, the gap remains between the user and the device. How should an AR/VR device be controlled, without the control mechanism being clumsy and get in the way of life? It would seem that gestures are the path to achieving this, and other research groups and industry vendors, for instance LeapMotion, are going in the same direction. Many gesture based control mechanisms today use cameras to track body or body part position. While cameras are getting smaller, a lot of preprocessing is needed in order to make sense of the visual data. Preprocessing in these scenarios is computation heavy and may need additional machine learning approaches to make computerised application possible. As a general day-to-day control mechanism cameras thus don't seem like a particularly good option.

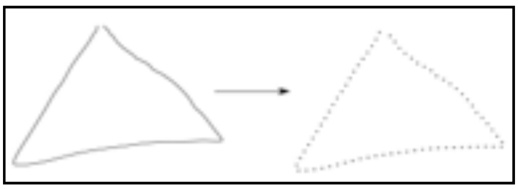
With this in mind, particularly the clumsiness of current control mechanisms, we set out to attempt to solve some of the problems in interacting with AR/VR environments. As Y. Hsieh from Helsinki University had already built a working prototype glove for testing and solving a similar case, we saw it as a natural direction to go to.

Gesture recognition

The algorithm chosen for the purpose was originally proposed by J. O. Wobbrock et al. in [1]. It is a simple '1$' recogniser that does not use complex signal preprocessing or machine learning. Rather, it consists of four stages of simple arithmetic on a drawn line and comparisons with 'templates'. At the core of the algorithm there are 'templates', which are point paths designated for any given gesture, against which the user input is compared. The comparison produces a score, and the comparison with the highest score yields the matched gesture.

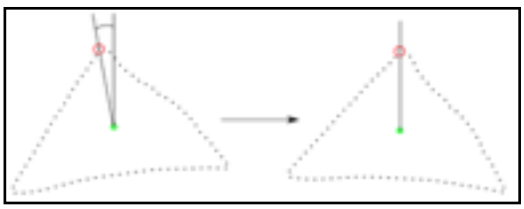
The algorithm functions as follows:

1. *Resample the point path:*

   The input device has a set sampling rate, thus movement speed has a clear effect on the number of points on the line at any given time. Additionally, the user might hold the device at a location yielding more points than necessary at a single location. Therefore the path is resampled so that each point is equally spaced out. The number of points on the resulting line can be varied, but the authors found any number of points between 32 and 256 to be adequate. Our implementation uses 64 points per line.

2. *Rotate once based on indicative angle*

   Each template is stored at a given angle and is a collection of ordered points. The user input must also be transformed to the same angle or as close as possible. The resampled point path is therefore rotated so that the angle that the first point and the centroid of the point path forming a line is set to 0°. This accelerates the finding of the optimal angle in the next step.

3. *Scale and translate*
   The point path is then scaled to a reference square so that it is the same ‘size’ as the templates. Scaling to a square does not hold aspect ratios and as a result draw rectangles cannot be distinguished from squares, for instance. Finally the points on the drawn path must be ‘set to zero’. In essence the reference point was chosen to be the centroid of the gesture, which is set to (0,0), and the other points moved respect to it.

4. *Find optimal angle for best score*
   Now the templates and user input has been preprocessed in the same manner. The algorithm now finds the average distance between points for each template, i.e. the path distance. During the process the preprocessed user input point path is rotated iteratively to find the lowest path distance for the given template. The minimum path distance is converted to a score, according to which a template is the closest to the drawn path.

The reader is encouraged to refer to the original paper for more details of the algorithm [1].

Prototype and implementation

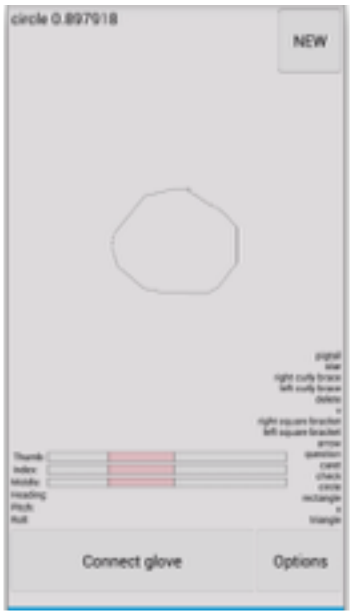
The prototype for the glove was built and supplied by Yi-Ta Hsieh, from Helsinki University. The right handed glove itself is built on top of an Arduino Pro mini. The prototype contains a 9-DOF IMU sensor (magnetometer, gyroscope and accelerometer), three bending sensors (one for each thumb, index and middle finger), three vibrator-acuators (one for each finger) and finally a bluetooth connection module for connecting it with a smart phone. The only modification to the glove was that the bend sensor reading was increased to support more fine grained control.

The glove prototype

The prototype recognition software was made for an Android device. It was implemented in Java and Processing languages. [Originally I attempted to modify the Arduino programming so that the glove could be registered as a generic HID device which could be augmented by a background Android app, but this proved to be impossible with the hardware at hand.] Source code is available on request, with attribution.

The Android app consists of the following parts:

1. The Processing core on Android. In order to be able to use Processing canvas and primitives the core of the Processing needed importing, (Java). [3]
2. The gesture recognition algorithm. This was implemented by Norman Papernick and imported from [2] with little modification (Java).



The prototype app allows drawing on screen

3. The user interface and drawing surface. (Processing)

The Android prototype gesture recognition software (henceforth prototype app) presents the user with a canvas that can be draw on. It responds to the user drawing using the handset touchscreen as well as input from the glove. To draw with input from the glove, the user connects the glove by pressing the ‘connect’ button. After successful connection, the glove sends data to the prototype app. The prototype app then uses the gyroscope data from the IMU sensor to determine the location of a pointer on the screen. The user can move the pointer by movements of the wrist, the prototype app does not use relative spatial data (x nor y) for controlling the pointer. To draw the user must select their preferred ‘activation position’ of the fingers, which is recorded per user. As the user returns their fingers to the activation position, the drawing mode is activated and a vibration pulse is sent to the actuators. When the user releases their fingers from the position the action stops and the recorded point path is injected to the gesture recognition algorithm. The result of the algorithm is then displayed on the screen.

Drawing with the glove. Level of bending in blue, activation threshold as red bar.

Other features of the prototype app, in addition to calibrating the user activation position, include:

- set the sensitivity for the pointer: the pointer location and sensitivity is determined by a moving average, which is at a maximum of 1 nit (the realtime position of the glove).
- set the activation threshold range: the activation position requires the fingers to be in a set position, which is recorded, but as the sensor readings shift and the position is never 100% accurate, there must be a range within which the app recognises the activation.
- set the pointer to return to ‘home’ position (centre of the screen) on activation. This may be useful when the user wishes to transfer from looking down at the screen to looking straight (horizontally) at the screen.
- manually set the ‘home’ position.
- activate calibration of the glove built-in magnetometer or accelerometer.
- Record new gestures and add previously unrecognised gestures as templates. Currently the user has no option of adding a previously unrecognised gesture as an additional template for a template that already exists.

Currently the glove input device has a major flaw which presents significant difficulties in activating and persisting the drawing mode. The readings from the bend sensors are not stable, i.e., they shift when the sensor is bent, to lower readings. This may be due to the fact that the sensors are quite old and worn.

Other options.

User evaluation

Any new system needs evaluation. Our system for gesture input was also evaluated in two user studies. The first user study was informal and attempted to discover bugs and major flaws in the prototypes (glove and app) before moving on to a more controlled study.

The first informal study included 5 individuals, 3 male and 2 female. The project was described briefly to the participants, most importantly how to draw with the glove. At this point calibration options included only magnetometer and accelerometer calibration, and the activation threshold was set to a predefined range. The study successfully pointed out some flaws and some other interesting phenomena. Firstly, and most importantly, the study revealed that the activation position is far from universal. In fact most participants (4 of 5) found that the position, in which they needed to bend their fingers to, was abnormal and physically stressful. Additionally, most participants had trouble keeping the activation position, as it was not a normal position, leading to incomplete gestures. Secondly it became clear that users have different orientations of their hand. The pointer, which helps the user in drawing, was often outside of the canvas, unseen, and the user needed to search for it. This happened with 4 out of 5 participants, where one had constant trouble with it. It should be noted that the pointer is a helper at this stage of the prototype and need not be present in other versions. These tow findings prompted the creation of the activation position and hand position calibration options so that the prototype app could be tuned to the user.

In other findings 4 of 5 participants did not find the movement of the wrist as an intuitive way of moving the pointer and drawing. They were constantly moving their hand in the relative x, y plane and not rolling and tilting their wrist. This lead to unrecognisable gestures, as the pointer hardly moved at all on the screen. Two of the five participants floated their hand above the screen, which became tiring. They said that they know that it is unnecessary but for some reason they feel like doing so. It was also clear from the study that all users had difficulty in replicating the gestures. The gestures that were easiest to replicate were circle and rectangle, other gestures were statistically not repeatable and most of the gestures were either not recognised at all or were recognised as other gestures.

The second more formal study had the following structure:

1. Basic information: gender, age group, education, devices used
2. Part 1 gestures: participant sees device screen, completes five of each circle, rectangle, triangle, star, curly brace in random fashion up to three attempts.
3. Part 2 gestures: same as part 1 except user does not see screen
4. NASA TLX and ending notes.

In total two people participated in the study. In the following tables, which summarise the study, positive equates to the gesture being recognised as the one intended, false positive means the gesture was recognised as another gesture and negative means the gesture was not recognised at all. Numbers are percentages.

Basic information:

	gender	age	education	PC / MAC	Tablet	smart-phone	gsm phone	laptop	smart TV
User 1	M	30-40	CS major	x	x	x	x	x	
User 2	M	20-30	CS major	x	x	x	x	x	x

Part 1 results:

	circle	rectangle	triangle	star	curly brace	Total
Positive	70	50	20	60	50	50
false positive	20	30	40	0	50	28
negative	10	20	40	40	0	22

Part 2 results:

	cirlce	rectangle	triangle	star	curly brace	Total
Positive	50	40	40	0	50	36
false positive	10	20	20	0	20	14
negative	40	40	40	100	30	50

Despite the low number of participants, some conclusions can be drawn from the results. First off, the participants demonstrated a big difference in their dexterity, which was reflected in their individual results. Therefore it would seem that the background from which the participants came from has little to do with the ability to use the device, which is expected. The results indicate that it was much easier to draw gestures while looking at the screen than not: 50% vs 36% positive results, 22% vs 50% negative respectively. While observing the motions of the gestures, we noticed that the participants tended to try to start moving their hand in the x,y relative position plane, rather than continuing using their wrist movement as drawing motion. This needs to be taken into consideration and implemented in the future version of the prototypes.

The results also give some indication about the difficulty of drawing and recognising different gestures. By far the easiest to draw and recognise was the circle, with 70% match rate and lowest overall non-match rate, 30%. As suspected, the rectangle is the next easiest to draw and recognise. To our surprise, the triangle was quite hard to draw and recognise. Another surprise comes from the curly brace, which was almost as easy to recognise as the rectangle in part 1 of the evaluation and its performance stayed the same in part 2. Overall it is clear that seeing the gesture path increases the probability of positive recognition. This is to be expected, but ideally would not be needed in a AR/VR setting.

NASA TLX:

	Mental demand	Physical demand	Temporal demand	Perfor-mance	Effort	Frustration
User 1	7	5	4	5	7	7
User 2	5	4	4	7	7	7
Average	6	4,5	4	6	7	7

The results from NASA Task Load Index indicate that the task was quite demanding and required quite a lot of effort. Even though the physical exertion is quite low and the user was not in a rush the effort put into accomplishing the task is high and resulting frustration high as well. High frustration is probably due to the user not being able to make a positively recognisable gesture and therefore 'failing' at the task. Taking into account the deficiencies of the prototype it could be argued that the participants performance was higher than what they perceived. The most concerning result is that the mental load was ranked high. Ideally using the glove or similar apparatus should be natural, intuitive. This may be corrected with simpler, yet distinguishable gestures, but the hypothesis needs to be tested.

The last part of the study, an open comment indicated that the star gesture was hardest to complete, which is also reflected in the results of the first and second parts. Additionally both participants clearly noted that the feedback (a short pulse on the vibration actuators) they received on activation was unclear. They said it was unreliable and did not know when to expect it. The source of the problem is not known at the time of writing. It is thus clear that the prototypes need more work and further evaluation for clearer results to be drawn.

Conclusions

As we enter the dawn of augmented and virtual realities we face a new problem, which is that of control. Our traditional mice and keyboards no longer work and new interaction mechanisms must be invented. Gesture recognition poses a promising domain. Gestures done in 2D spaces have been successfully made use of, but moving the domain knowledge to 3D is still an incomplete task. We have exhibited a rudimentary system which is able to recognise gestures drawn in mid-air. The system used minimally preprocessed data from a gyroscope planted in a wearable glove and projected it to a 2D plane. The system can be augmented to include other dimensions or combinations thereof to achieve more complex gestures. The user evaluation indicated that more work needs to be put towards making the recognition more reliable, first of all adding the x, y relative position as a component with the use of data from an accelerometer. The activation of the gesture recognition mode plays the most crucial part in gesture recognition and user satisfaction. Perhaps the sensors measuring the bending of a finger are not adequate in this respect. The prototypes, nevertheless, indicated that simple data from low power sensors may be enough in controlling AR/VR scenarios, in lieu of complex camera settings and high compute machine learning algorithms.

References

[1] Wobbrock, Jacob O., Andrew D. Wilson, and Yang Li. "Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes." *Proceedings of the 20th annual ACM symposium on User interface software and technology*. ACM, 2007.
[2] http://depts.washington.edu/aimgroup/proj/dollar/others/java.zip
[3] http://andymodlaphotography.blogspot.fi/2015/07/processing-sketches-using-android-studio.html