Interim Report: Mixed Reality Tabletop War Game Assistant

COMP3003

Nathan Burgess psynb7 20363169

BSc Computer Science

Table of Contents

1. Introduction and Motivation 3
1.1. Related Work 4
1.1.1. Previous Mixed Reality Tabletops 4
1.2. Project Description 7
2. Methodology and Design 7
2.1. RFID 7
2.1.1. Machine / Deep Learning 9
2.1.2. ARKit 9
2.1.3. Computer Vision without neural networks 9
3. Methodology, Design and Implementation 10
3.1. Design - Talk about Open CV, why the cameras are the way they are and why I chose to use python as well as the library for the digital part. 10
4. Progress 14
4.1. Project Management 14
4.2. Contributions and Reflections 18
4.2.1. LSEPI 18
5. Bibliography 18

1. Introduction and Motivation

Tabletop Wargaming is a popular hobby but, with a high barrier to entry, it remains niche and inaccessible to many. The rules to tabletop wargames can be complex and difficult to learn. This can be daunting for new players putting them off the hobby as well as causing arguments between seasoned players over different rules interpretations.

The most popular wargaming systems are produced by *Games Workshop* [1]. One of their more popular systems, *Warhammer 40k*, has a core rulebook of 60 pages [2] and the simplified version of another game system, *Kill Team*, is a rather dense three page spread [3]

Kill Team is a miniture wargame known as a "skirmish" game. This means the game is played on a smaller scale with only /20 minitures on the table at one time. The aim of the game is to compete over objectives on the board for points. Each player takes turns activating a miniture and performing actions with it. These actions can involing moving to another location, shooting at an enemy or capturing an objective. The game uses dice to determine the results of your models engaging in combat with eachother.

Figure 1: An example of the *Kill Team* tabletop game using the *Gallowdark* terrain

Video games help on-board new players by having the rules of the game enforced by the game itself. This project aims to bring this experience to tabletop wargaming, specifically the *Kill Team Lite* [3] system using the *Gallowdark* setting. This is because the *Kill Team Lite* rules are publically available from *Games Workshop's* website and it is designed to be played on a smaller scale to other wargames, making it a good candidate for a proof of concept. As well as this, the *Gallowdark* [4] setting streamlines the terrain used and removes verticality from the game, making implementation much simpler.

Developing a system that can accurately track the position of minitures and terrain in a *Kill Team* board would allow for the creation of various tools to assist in the game. In this case a digital game helper would move the burden of rules enforcement and onto the system, allowing players to focus on the game itself or allow players to make more informed game decisions by being able to preview the options available to them. Model tracking could also be utilised to record a game and view a replay of it or be utilised for content creation to create accurate board representations to viewers with visual affects.

1.1. Related Work

Previous attempts at created mixed reality tabletops fall into 2 categories. augmented / virtual reality or physical to digital. This will take a focus on a physical to digital approach as being able to use the minitures you built and painted yourself is an important part of the wargaming hobby.

Physical to digital systems use a physical game board and some detection system to find the pieces on the board. These can be further split into two categories, detection from above and detection from below.

1.1.1. Previous Mixed Reality Tabletops

Some companies, such as *The Last Game Board* [5] and *Teburu* [6], sell specialist game boards to provide a mixed reality experience.

The Last Game Board achieves this through utilising the touch screen to recognise specific shapes on the bottom of minitures to determine location and identity. *The Last Gameboard* is 17″x17″, as a result the number of game systems which are compatable is limited. However, you can connect multiple systems together. The drawback of this is the pricepoint for the system is rather high, with boards starting at ~$350.

Figure 2: The Last Game Board touchscreen tabletop system [5]

Teburu [6] instead takes an RFID based approach, providing a base mat that allows you to connect squares containing RFID recievers and game pieces containing an RFID chip. *Teburu* connects to a tablet device to provide the digital experience as well as to multiple devices for indvidual player information. *Teburu* games allow for game pieces to either be in pre-determined positions or within a vague area i.e. within a room.

Figure 3: The Teburu Game System [7] showcasing *The Bad Karmas* board game. The black board is the main game board. The squares above connect to the board below to transmit the RFID reader information back to the system for display.

An RFID based approach is also used by *Steve Hinske* and *Marc Langheinrich* in their paper [8] which places an antenna grid below the game board to detect RFID chips in pieces. This allowed them to find what chip is in range of what antenna, allowing them to find the general location of a game piece. This worked particularly well for larger models where you could put RFID chips far away from eachother on the model. Using the known positions of the chips and dimensions of the model combined with which antenna said chips are in range of allows you to determine an accurate position of each model. They also go into alternate RFID approaches which will be discussed later when outlining the chosen methodology.

Figure 4: An example of Steve Hinske and Marc Langheinrich's approach depicting the antenna grid, RFID tags and physical model alongside the computer's prediction of the model's position

Surfacescapes [9] is a system developed in 2009 by a group of masters students at Carnegie Mellon as a university project. *Surfacescapes* uses a *Microsoft Surface Tabletop* (the product line was rebranded to *PixelSense* in 2011). This uses a rear projection display, and 5 near IR cameras behind the screen [10]. This allows the *PixelSense* to idenify fingers, tags, and blobs touching the screen using the near IR image. *Surfacescapes* utlises this tag sensing technology to track game pieces using identifyable tags on the bases of minitures.

Figure 5: An example of *surfacescapes*' in use on the *Microsoft Surface Tabletop*[11]. The position of the models have been tracked by the system and outlined with a green cricle.

Foundry Virtual Tabletop [12] is an application used to create a fully digital *Dungeons and Dragons* tabletops. These can either be used for remote play or in person play using a horizontal TV as a gameboard. *Foundry VTT* allows for the creation of modules to add new functionality to your virtual tabletop. One such module is the *Material Plane* [13] module which allows the tracking of physical minitures on a TV game board. This functions by placing each miniture on a base containing an IR LED with an IR sensor then placed above the board. This can be configured to either move the closest "virtual" model to where the IR LED is or (with some internal electronics in the bases) can be set up to flash the IR LED in a pattern to attatch different bases to specific models. An indicator LED is present to show when the IR LED is active.

Figure 6: An example of one of the *Material Plane* bases. A miniture would be attatched to the top. [14].

1.2. Project Description

This project aims to create a system that can track tabletop minitures, in a game of *Kill Team Gallowdark*, using only materials easily accessable to the average miniture wargamer. Then, utilise this system to implement a "helper" program for the game, providing a digital representation of the physical game state as well as rules enforcement.

As a result, the project can be broken down into two main goals.

1. Detection of the models and terrain to create a virtual representation of the game board.
   a. The position of the miniture must be tracked accurately.
   b. The system must be able to identify between different minitures.
   c. The system must aim to be un-invasive to the minitures.
   d. The system must be ambivalent to a model's shape and colour.
   e. The system must be able to complete this task whilst being accessible to the average miniture wargamer.
2. Implementing the game logic in the virtual board to guide players through the game.
   a. Allow users to select a model and a subsequent action (normal move, shoot, dash, capture objective).
   b. Calculate the distance a model can move and display this on the virtual board.
   c. Account for terrain that blocks movement.
   d. Calculate the line of sight between the selected model and opposing models then display this on the virtual board.
   e. Account for terrain that blocks line of sight.
   f. Display information about the selected model's odds to hit a target.

2. Methodology and Design

Finding a method to detect and find the positions of a minitures on a game board, while not obstructing the game, is a challenge. The majority of the work done so far has been in researching and designing different approaches to this problem. This section will outline the different approaches that have been considered.

2.1. RFID

An RFID approach is the somewhat obvious solution. This would involve embedding RFID chips underneath the bases of the minitures. Then some method of reading these chips would then need to be embedded ether within or underneath the gameboard. There are a number of different approaches that could be taken to locating RFID chips which have been outlined by *Steve Hinske* and *Marc Langheinrich* [8] in their work on a similar project.

RFID solutions would require either an antenna grid underneath the gameboard or multiple individual RFID readers. This would be a viable option as hiding an antenna grid below a board is a relatively simple and unobstructive task. The same goes for hiding RFID readers beneath a table.

The main drawback of RFID is that a reader (at its core) can only detect if a chip is in range or not (referred to as "absence and presence" results). Due to this, some extra methodology would need to be implemented to determine the position of a chip.

One approach utilises increasing the range of an RFID reader to its maximum and then subsequently reducing the range repeatedly. Using the known ranges of the readers it would be possible to deduce

which tags are within range of which reader - and subsequently deduce from which ranges it is detectable in the position of the RFID chip. This approach could in theory provide a reasonably accurate position of the RFID chip. However, this approach would take a longtime to update as each RFID reader would need to perform several read cycles. Combined with problems caused from interference between the chips / readers, the fact that being able to vary the signal strength of an RFID reader is not a common feature and the need for multiple RFID readers, this approach does not meet the requirements for this project.

Another approach utilises measuring the signal strength recieved from an RFID chip and estimating the distance from the reader to the chip, known as received signal strength indication (RSSI) . This approach is much quicker than the previous method needing only a single read cycle to determine distance. Most modern day RFID readers can report RF phase upon tag reading. However, current RSSI methods have an error range of ~60cm caused by noise data, false positives / negatives and NLOS (non-line of sight) [15]. This won't work for this project given than the board size is 70 x 60 cm.

Triliteration is a process in which multiple receivers use the time of arrival signal from a tag to detmine it's position. This suffers from similar problems to other RFID methods in that it produces an area in which the tag could exist within - as opposed to it's exact position. Combined with the need for 3 RFID receaders, this approach fails to be accessable to the target audience.

The approach previously mentioned in *Steve Hinske* and *Marc Langheinrich*'s paper, which utilised an antenna grid below the gameboard, seemed promising.

Figure 7: An example from *Steve Hinske* and *Marc Langheinrich*'s paper [8] describing their overlapping approach. The black square represents the RFID tag whilst the grey circles show the read area of each RFID reader. The intent is to use which readers are in range of the tag to determine an "area of uncertainty" of the tag's location.

This approach worked rather well for *Steve Hinske* and *Marc Langheinrich* as they were attempting to do this for *Warhammer 40k* a game played on a much larger space typically with larger models. As a result they were able to put RFID tags across larger models, such as vehicles, and utilise the known position of each tag relative to the model and the estimated position from the RFID readers to determine not only an accurate position but also orientation. Unfortunatly for this project the size of the minitures in *Kill Team* would prohibit me to using one tag or two in very close proximity.

A potentially valid method utilising RFID could be to use the approach outlined by *Zhongqin Wang, Ning Ye, Reza Malekian, Fu Xiao, and Ruchuan Wan* in their paper [16] using first-order taylor series approximation to achieve mm level accuracy called *TrackT*. However, this approach is highly complex and can only track a small amount of tags at a time.

2.1.1. Machine / Deep Learning

Modern object tracking systems are often based on a machine learning or deep learning approach. In this case a classifier model would be used to identify each unqiue miniture on a team, and then subsequently locate it within the game board. The biggest drawback to this approach is the amount of training data needed for each class in a classifier. According to *Darknet's* documentation (a framework for neural-networks such as YOLO) [17] each class should have 2000 training images.

Since each user will use their own minitures, posed and painted in their own way, they would have to create their own dataset for their minitures and train the model on them each time. As a user's minitures are likely to follow a similar paintscheme, ML classification could potentially struggle to indentify between minitures from top down and far away if not enough training data is supplied.

2.1.2. ARKit

Apple's ARKit supports the scanning and detection of 3D objects [18] by finding 3D spatial features on a target. Currently any phone running IOS 12 or above is capable of utilising the ARKit. In this system, you could scan in your models, then use the ARKit to detect them from above. This information could then be conveyed to the main system. This could also allow for the system to be expanded in the future to use a side on camera as opposed to just top down detection, allowing for verticality within other *Kill Team* game systems. Combined with certain Apple products having an inbuilt LIDAR sensor (such as the *iPhone 12+ Pro* and *iPad Pro* 2020 - 2022), which further enhances the ARKit's detection, this could be a viable approach.

Figure 8: Registering an an object in ARKit [18]

After some testing utilising an iPad Pro with a LIDAR scanner, some drawbacks were found with this approach. The object detection required you to be too close to the models detect them, and when being removed and re-added ARKit either took a long time to re-locate a model or failed to do so at all.

As a result I won't be using this approach for the project. Allthough, for future work, this could be an interesting option to explore allowing you to implement AR features such as highlighting models on your phone or tablet or displaying information above the model.

2.1.3. Computer Vision without neural networks

After considering all the options above, the best method found was using computer vision techniques.

One technique is blob detection. A blob is defined as a group of bright or dark pixels contrasting against a background. In this case the minitures would be the blobs.

Utilising blob analysis would allow locating general position of minitures (as they would very clearly stick out from the background) but getting their exact center may then prove difficult (which would be needed for calculating movement and line of sight). Combined with the addition of terrain adding clutter to the image and the minitures potentially being any colour, this approach could prove difficult to implement in the given timeframe.

The computer vision method explored the most was utilising coloured tags to identify minitures.

The use of coloured tags would also mean there would be access to specific measurements allowing for an accurate location to be discerned. However, some external modifciation to the the gameboard or minitures would need to be made. The challenge here is finding a way to do this without obstructing the flow of the game or the models themselves.

3. Methodology, Design and Implementation

3.1. Design - Talk about Open CV, why the cameras are the way they are and why I chose to use python as well as the library for the digital part.

The general design of the system would placewebcam or camera above the gameboard. Minitures would have easily recognisable tags attatched to allow for easy identification via computer vision.

The tag would need to be unobstructive to the game (big enough to be seen by the camera but small enough as to not get in the way of the game whilst also not looking too "out of place"). This ruled out sticking QR codes ontop of minitures, or attatching them to wires above the minitures. This meant a unique design was needed.

Before we start trying to find tags we should first perform some viewpoint / perspective correction. Our end goal is to get a birds eye view of the gameboard, however a camera placed above the board will produce a slightly distorted view of the gameboard. This process would involve finding some set reference points on the board, in this case the corners, and using these to perform a perspective transform creating a top down view of the image. This means that when the tags are located we can provide their location directly to the gameboard display.

The detection problem can be split into two components: locating and identifying the miniture. Tackling the location problem first seemed like the most sensible approach to begin with before moving onto identification.

In the end it was chosen to 3D print a small rim that could fit around the base of a miniture. This rim would be a high contrasting colour to the background and miniture. The benefits of 3D printing the rim is that the colours and dimensions are customizable to the user. Miniture wargamers paint their models in a variety of different ways, so having a rim they can decide the colour of themselves keeps the system accessable to the target audience and open to many different miniture sizes and colours.

Figure 9: An example of the basic contrasting rim design.

Some important observations here are that models may poke over the edge of the rim and models will block the rim due to parallax, so a rim detection system will need to find a full circle from only a partial amount of visable rim. When terrain is introduced this will cause further issues The proposed fixes for this will be discussed later.

The next step was to find a way to detect the rim. The best solution found for this was *openCV* [19]. An alternative image processing suit would have been to utilise *MATLAB*, but *openCV* can be used directly within *Python*. As a result, the final system could be written entirely within *Python* signficantly streamlining the development process without needing to connect MatLab with an external program for the game board logic / display.

This led to the development of a processing method which will locate the center point of circles of a specific colour in a provided image.

The image processing steps taken:

1. Sort the image to only contain the colour of the rim (in this case yellow).
2. Greyscale the image.
3. Apply a blur to the image (this helps to remove noise from the image).
4. Apply hough circle detection [20] on the processed image.
5. Draw the detected circles onto the original image.

It is important to note that hough circle detection can easily mistake two close but separate circles as one circle. As a result of this, the hough circle detection function in *openCV* has several arguments that will need to be tweaked as the system is developed.

Through some testing we were able to locate singular circles even with the presence of terrain pieces blocking parts of the rim.

Figure 10: Circle detection on a gameboard with example terrain.

Parallax is a problem that still needs to be addressed. Through testing it was found that using a standard gameboard size, the parallax was too great to expect to locate a rim. As a result, we have settled on using a two camera solution. This solves the parallax problem by having each camera watch a different half of the gameboard.

However, one problem with this approach is that calibration will be needed to ensure that the overlap between the cameras does not cause issues. This could be achieved by either calibrating the cameras to the game board, as it will be of a known size, by placing tags on each corner of the board. Alternatively the middle of the board could be marked with a piece of tape, or tags on the edges. This should then allow for the two images to be stitched together accuratly.

Due to the nature of this project being aimed at following the *Kill Team Gallowdark* ruleset, the terrain pieces used are placed along a grid. This prevents a situation where a terrain piece is very close to the edge of a board with a miniture placed behind resulting in the miniture being blocked from view.

Once we have located a model we need to identify it. In our implementation of *Kill Team* we will assume that each player will have up to 14 models on their team (known as "operatives"). As a result we will need to create an encoding capable of representing at least 28 different options.

Another rim will be placed around the bright contrast rim from before. This rim will be split into 6 sections. The first section will indicate the starting of the encoding and will use a unique colour. The other 5 sections will use 2 different colours to represent 1 or 0. Once a rim has been located, the system will then determine the orientation of the encoding circle. As the size is known, we can then rotate around the image of the encoding circle reading each section to eventualy get a binary number. Each binary number will then represent a unique model on the board.

Figure 11: An example of what the encoding circle could look like. The red section indicates the start of the encoding. The system would be able to determine which direction to read in based on the start encoding section being in the bottom or top half of the image.

Another encoding method that was considered was to use one rim for both detection and identification. By using one rim of known colours, the image could be split into 3 images each containing only the colour of each segment. Each image would then be converted to black and white and recombined to create a singular image with the segments connected to create a full circle. After circle detection has been performed the system could then do the normal encoding reading. This approach would allow the rims of the minitures to be significantly smaller and only require three contrasting colours.

One problem faced by this approach is that vision of the entire circle is needed to read the encoding, as a result if it is partially obscured then the system will be unable to identify the model.

This could be counteracted by splitting the encoding into quarters around the rim, so that as long as 1/4 of the encoding is visable, the system can correctly identify it. We can also exploit *Kill Team* being a turn based game and simply compare the current game state with the previous game state to identify which pieces have been moved. A combination of active tracking and being able to determine which piece has moved, should allow for a robust system.

Terrain detection will be performed using two possible solutions, the user can either select pre-determined terrain set ups or the system can detect the terrain pieces.

The current approach for terrain detection is to use QR codes on the top of each terrain piece. In *Gallowdark* the size and shape of each terrain "module" is already known. As a result, the QR code can be used to determine the type of terrain and orientation. Then, since the terrain type has been established we already know the size and shape. So we can then fill it in on the virtual board.

Once the system has correctly identified and located each model and the terrain, a digital representation of the gameboard can be created.

Figure 12: A proposed GUI for the game helper.

In Figure 12 the red semi-circle represents the valid area of movement for a selected operative. This can be calculated by splitting the game board into a very small grid, we can then apply a pathfinding algorithm (Dijkstra's, A* etc) to the grid from the operatives starting point. We can then calculate all grid positions reachable by using a movement value less than or equal to the current operatives available movement characteristic.

Red circles around opposing opperatives show they are in line of sight. This would be calculated using simple ray casting. A white and red circle indicates a selected opposing opperative. This is to calculate the statistics on the right hand side of the screen to be specific to the opposin opperative.

The information for the selected operative is displayed below the gameboard. A user can select the action they have performed / want to perform on the right hand side.

The GUI will be built in *PyQt* [21] which is a python wrapper for the *Qt* framework. One potential strech goal is for the GUI and the detection system to be seperate entities. For example, the detection system should be able to detect and track pieces without input needed from the virtual gameboard i.e. which model is selected. This would allow for other detection systems to be swapped out in the future or for other applications to be developed using the detection system.

4. Progress

4.1. Project Management

As per the inital Project Proposal the original goal for the first semester was to have ring detection and terrain detection completed. Currently, simple ring detection has been completed. This still needs to be expanded to include identification, though work has begun on this. Terrain detection has been moved backwards to allow for more time to make a good detection system for the minitures.

Choosing to tackle the ring detection first was a good choice. This allowed for me to encounter larger issues and provide solutions to them whilst still in the research stages on the project. This has allowed

for me to develop my computer vision skills and take approaches I would not have otherwise considered. For example, using a singular rim of multiple colours and combining the images to create a full circle for identificaton.

Supervisor meetings were held every week to check for blockers or advice on reports. Uniquely, our supervisor meetings were conducted as a group with the other two dissertation students my supervisor had. As all three of us were working on tabletop game based projects we were all knowledgable in the area. This meant we could provide feedback or ideas for eachothers projects from a student perspective.

I found the GANNT chart to be unhelpful in managing the project. GANNT charts work well in well structured work environments.However, due to the nature of this semester having a large amount of other courseworks and commitments that needed to be balanced. An agile approach was much more effective, doing work when time was available. The nature of development work being much more effective when done in long, uninterupted sessions meant that, with this semesters courseworks and lectures being very demanding at 70 credits, being able to find long swathes of uninterupted time was very difficult between lectures, courseworks, tutorials etc. As a result, development was done in smaller, more frequent chunks which were not as effective as longer, less frequent chunks. I expect this to change next semester when the workload decreases down to 50 credits.

A much prefered method, that I will likely go ahead with, is Kanban. I personally found most use from the GANNT chart in that the project is naturally broken up into sub tasks. This type of subtasking when combined with a less structured agile approach is something that Kanban works really well at and have found effective in the past in GRP.

In the interests of being able to show current progress in comparison to previous I have included an updated GANNT chart (Figure 13) along the preivous one (Figure 14).

Looking at the time remaining I will focus on getting the main parts of the system functional. These are: model detection and tracking, terrain detection, virtual game board representation, movement preview and line of sight preview. If there is time available then I will aim to also implement the flow of the game (breaking it down into each phase, providing guidence of what to do in each phase, statistics etc). The most technically interesting part of this is the virtual game board and tracking technology. So placing a focus on having them work fluidly is more important to the dissertation.

Figure 13: The Updated GANNT chart

Figure 14: The original GANNT chart

4.2. Contributions and Reflections

Something I underestimated was the amount of time needed to research and determine which method should be used for model detection. A large amount of time was spent determining the viabiity of different approaches that may make the system more robust and easier to produce. As a result of this, development started later and took longer than was expected.

The biggest impact on this project was needing to get an extension on other coursework deadlines. This extension had a knock on effect of other courseworks which took priority over this project. The timings alloted in the GANNT chart did not take into account extensions being needed.

However, I am happy with the progress made so far. Having completed the methodology to find the circular tags even when partically obstructed is very promising to the viability of the chosen approach. As a result, I believe that a solid strategy has been chosen for model detection and identification and the work done so far has successfully laid a good foundation for the project.

4.2.1. LSEPI

The main LSEPI issue you could see here is copywright issues as *Kill Team* is a copywrighted entity owned by *Games Workshop*. The way this project will handle this is by having the system implement the *Kill Team Lite* ruleset previously mentioned. This is publically available published by *Games Workshop*. One issue is that different "*Kill Teams*" (groups of operatives) will have different and unique rules as well as their own statistics pages. These are copywrighted and won't be able to be directly implemented. As a result of this, this project will implement basic operatives with fake statistic pages based off of the publically published *Kill Team* information. If this proves to be problematic then the game rules will be based off a very similar system *Firefight* [22]. This is published by *One Page Rules*, who produce free to use, public miniature wargame systems.

5. Bibliography

[1] Games Workshop, "Games Workshop Investor Relations Statement". Oct. 17, 2023. [Online]. Available: https://investor.games-workshop.com/our-history

[2] Games Workshop, "Warhammer 40k Core Rules". Jun. 02, 2022. [Online]. Available: https://www.warhammer-community.com/wp-content/uploads/2023/06/dLZIlatQJ3qOkGP7.pdf

[3] Games Workshop, "Kill Team Lite Rules". Aug. 16, 2022. [Online]. Available: https://www.warhammer-community.com/wp-content/uploads/2022/08/ekD0GG2pTHlYba0G.pdf

[4] Games Workshop, "Killzone Gallowdark". Accessed: Dec. 12, 2023. [Online]. Available: https://www.warhammer.com/en-GB/shop/killzone-gallowdark-2023

[5] Last Gameboard, "The Last Gameboard Kickstarter". Accessed: Dec. 12, 2023. [Online]. Available: https://www.kickstarter.com/projects/gameboard1/gameboard-1

[6] Teburu, "Teburu". Accessed: Dec. 12, 2023. [Online]. Available: https://teburu.net/#home

[7] "Teburu: a sneak peek". Accessed: Dec. 13, 2023. [Online]. Available: https://www.youtube.com/watch?v=5paNoKA4b5A

[8] Steve Hinske and Marc Langheinrich, "An RFID-based Infrastructure for Automatically Determining the Position and Orientation of Game Objects in Tabletop Games". Jun. 05, 2007. [Online]. Available: https://vs.inf.ethz.ch/publ/papers/hinske-pg07-rfidtabletop.pdf

[9] Whitney Babcock-McConnell, Michael Cole, Stephen Dewhurst, Bulut Karakaya, Dyala Kattan-Wright, and Maokai Xiao, "Surfacescapes". Accessed: Dec. 13, 2023. [Online]. Available: https://www.etc.cmu.edu/projects/surfacescapes/index.html

[10] Microsoft, "Microsoft Surface 1.0 documentation". Accessed: Dec. 13, 2023. [Online]. Available: https://social.technet.microsoft.com/wiki/contents/articles/8017.microsoft-surface-1-0-sp1-getting-started-guide-hardware-overview.aspx#Tabletop

[11] Abhay Buch, "With Surfacescapes, Dungeons & Dragons becomes high-tech". Accessed: Dec. 13, 2023. [Online]. Available: http://thetartan.org/2010/4/5/scitech/surfacescapes

[12] Foundry Gaming LLC, "Foundry VTT". Accessed: Dec. 23, 2023. [Online]. Available: https://foundryvtt.com/

[13] Material Foundry, "Material Plane". Accessed: Dec. 23, 2023. [Online]. Available: https://www.materialfoundry.nl/

[14] Material Foundry, "Material Plane Github". Accessed: Dec. 23, 2023. [Online]. Available: https://github.com/MaterialFoundry/MaterialPlane/wiki/Beta-Hardware-Guide

[15] "Robust and Accurate Indoor Localization Using Learning-Based Fusion of Wi-Fi RTT and RSSI". 2022. Accessed: Dec. 27, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9002808/

[16] Zhongqin Wang, Ning Ye, Reza Malekian, Fu Xiao, and Ruchuan Wang, "TrackT Accurate tracking of RFID tags with mm-level accuracy using first-order taylor series approximation". Dec. 15, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870516302736

[17] "YOLOv4 / Scaled-YOLOv4 / YOLO - Neural Networks for Object Detection (Windows and Linux version of Darknet)". Accessed: Dec. 28, 2023. [Online]. Available: https://github.com/AlexeyAB/darknet

[18] Apple, "ARKit". Accessed: Dec. 27, 2023. [Online]. Available: https://developer.apple.com/documentation/arkit/arkit_in_ios/content_anchors/scanning_and_detecting_3d_objects

[19] G. Bradski, "The OpenCV Library". 2000. Accessed: Dec. 12, 2023. [Online]. Available: https://github.com/opencv/opencv

[20] G. Bradski, "Hough Circle Transform". Accessed: Dec. 27, 2023. [Online]. Available: https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html

[21] Riverbank Computing Limited, "PyQt". Accessed: Dec. 31, 2023. [Online]. Available: https://riverbankcomputing.com/software/pyqt/intro

[22] One Page Rules, "Firefight". Accessed: Dec. 31, 2023. [Online]. Available: https://www.onepagerules.com/games/grimdark-future-firefight