Accuracy of the Hololens 2's infrared cameras in the context of surgical navigation

Omar Hussein
Supervisor(s): Pierre Ambrosini, Ricardo Marroquim
EEMCS, Delft University of Technology, The Netherlands

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

Patient and instrument tracking are fundamental parts of surgical navigation systems. Traditional surgical navigation systems rely on stationary cameras for tracking and stationary screens for presenting information. An increased mental load is exerted by surgeons as they switch focus between the surgical site and the presented data. By projecting the required data on the patients anatomy, augmented reality surgical navigation would allow surgeons to navigate seamlessly during surgical procedures. In this paper, an augmented reality tracking algorithm for the Hololens 2 will be presented and tested against state of the art optical tracking. The algorithm is based on the use of reflective markers, which are extensively used in medical applications. An accuracy of 1.04mm was found in optimal conditions using the presented method. The presented tracking method is less accurate and less consistent than state of the art optical tracking methods with a minimum median accuracy of 25mm.

1 Introduction

Surgical navigation systems (SNS) allow surgeons to precisely track the location and orientation of surgical instruments throughout a procedure. It allows surgeons to project the positions of instruments on preoperative data, serving as guidance system during operations. By doing so, SNS support surgeons in locating specific anatomical structures and avoiding areas of risk, by suggesting the optimal route to reach the identified location. Modern SNS have become increasingly reliant on infrared technologies (IR). By using stereoscopic cameras, that emit IR light, significant structures such as reflective markers can be tracked in real time[1] [2]. The respective tracking software must be used based on the hardware configuration to show the instruments in their anatomical position. SNS have advanced significantly over the last three decades and have been praised for their direct contribution to the transformation of surgical interventions into safer and less invasive procedures[3]. However, conventional surgical navigation systems do not come without challenges. The need of a fixed screen in conventional SNSs requires the surgeon to continuously switch focus between the surgical site and the screen. This adds difficulties with regards to the surgeons hand-eye coordination and the perception of the 3D anatomy[4]. As such there is still room for improvement in the field of surgical navigation.

One proposed alternative, to the use of fixed screens, is augmented reality (AR) headsets[6]. AR technologies have advanced significantly over the past couple of years and AR headsets have become commercially available. AR technology allows for the superimposition of hidden structures onto a visible surface. Using AR headsets would remove a surgeon's need to frequently shift focus between the patient and the screen during operation. Thus, reducing their mental load as well as enhancing their hand-eye coordination[7]. To maintain the correct projection over time, however, the patients movements must be tracked accurately. The movement of the surgeon must also be compensated for as well as the positions of the tools. Marker based tracking is a great fit for use in AR SNS, as it has already been proven effective in conventional SNS. By utilizing the IR cameras available in AR headsets, marker based tracking methodologies can be implemented in an AR SNS.

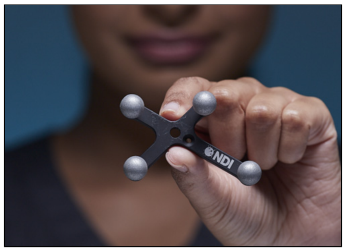


Figure 1: NDI optical marker with 4 reflective spheres used in medical practices [5]

The aim of this research is to examine the feasibility, effectiveness and accuracy of using AR headsets as a SNS. Specifically, in the context of image-to-patient registration and tool optical tracking. This will be done by building the basic components and functionalities for an augmented reality surgical navigation system using the HoloLens 2, a commercial AR headset. In this work a marker based tracking system will be created and evaluated for accuracy to solve the following question: "How accurate is 3d optical tracking using infrared cameras of the HoloLens in comparison with state-of-the-art approaches?". The tracking method, would allow for the relative localization of surgical instruments and the surgical site in real time[7]. Kunz et al.[8], have shown that the Hololens 1 was able to track IR markers with a tracking accuracy of 0.76mm. Moreover, the results presented by Kunz et al. [8] were recorded in a controlled environment where a complete depth map was provided for every frame and fixed translations of the marker were recorder with no rotations. Gsaxner et al.[7] also presented a tracking solution using the Hololens 2. Their work was based on the use of the stereo grey scale cameras in combination with an external light to emulate the intensity signature of using an infrared camera. The work of Gsaxner et al.[7] produced results of 1.70mm accuracy. Building upon their work, using the HoloLens 2, will show whether it is possible to achieve better more consistent tracking results, in a real life context with random movements and inconsistent depth maps using the IR cameras.

The methodology and approach used in this research is explained in Section 2. Section 3 contains the results produced from the methodology introduced in Section 2. A reflection on the ethical aspects of the research and its reproducibility is given in Section 4. Finally a discussion along with the most interesting findings and possibilities for future work is presented in Section 5 and Section 6, respectively.

2 Methodology

In order to track IR markers, an IR tracking algorithm was created. The algorithm processes the input images produced by the Hololens and outputs a set of positions for each IR marker at a given timestamp. The algorithm contains several steps that allow for the calculation of the 3D positions of each reflective sphere. In this section a detailed explanation of each step is given and an example of the input and output results is shown. Subsection 2.1 provides an overview of the provided resources at the start of the research. Subsection 2.2 presents a general overview of the methodology. Subsection 2.3 goes into detail on how the algorithm was built.

2.1 Data Set and Utilities

As input a set of different camera and sensor data, produced from the HoloLens 2, is provided. The ground truth 3D positions of the optical markers are also given for the evaluation of the tracking algorithm. The ground truth values are provided using the Polaris Vega VT optical tracker from NDI[5] which provides sub-millimeter accuracy. Since the proposed research question investigates the usage of IR cameras, only 2 sources of input are relevant. Those are the Articulated Hand Tracking (AHAT) and Long Throw (LT) cameras. The AHAT camera is used for near-depth sensing used for hand tracking, while the LT camera is used for far-depth sensing used for spatial mapping[9]. The method used for this research applies to both input feeds. However, examples produced from processing the AHAT input feed will be presented as the results are more favorable. The AHAT camera produced better results due to its higher resolution frames, higher frame rates, as well as its focus on closer positions. Moreover, each camera provides a depth map for each frame using IR sensors. The depth maps are not accurate enough to use independently. Different utilities were provided to allow for faster development given the time constraints. Utilities for easy loading and saving of the dataset, a GUI for visualizing the data, as well as utilities for projecting and unprojecting points to different spaces.

2.2 Method Overview

To achieve real-time tracking, each camera frame must pass through a processing pipeline. Figure 2 shows a diagram of the processing stages. The input camera frame is colorized such that high IR values have a red color while low IR values have a blue color. The high IR values produced from the spheres make them easier to isolate from the background. Consequently, the first step of processing is to isolate the

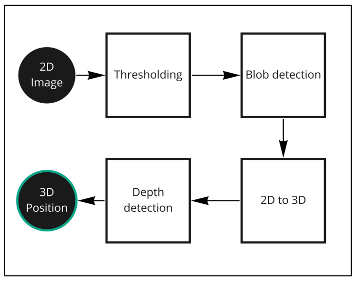
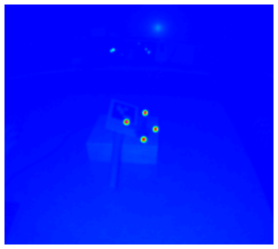


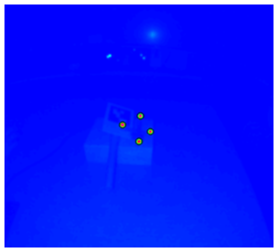
Figure 2: Processing pipeline

markers from the background. This can be done by binarizing the image. The image is binarized by setting every value



(a) AHAT Input Frame

(b) Binarized Frame



(c) Blobs highlighted on input frame

Figure 3: Sample Frames

above a specific threshold to the maximum value and everything below the threshold to zero. The appropriate threshold can be found by using a high percentile value. Blob detection methods are used to detect regions in a digital image that show defined characteristics, such as a specific area measurement, circularity or color. Thus, once the image has been binarized, blob detection can be used to retrieve the pixel position, keypoint, of a reflective sphere. Blob detection is thus applied to the binarized image in Figure 3b. The input frames with the spheres circled using a black outline can be seen in Figure 3c. To retrieve the actual 3D positions, the 2D positions must be unprojected. Projection is the term used to convert a point in 3D to a point in 2D for display on a screen. It is done by applying a specific set of transformations to the point based on its orientation, position, and the physical properties of the camera used. Usually an intrinsic matrix and extrinsic matrix are multiplied to the position of a point in space to project it and the inverse is done to unproject the point. Instead, the HoloLens 2 facilitates a lookup table that maps each pixel to a 3D position bound to the plane (x, y, z=1). As such the distance of the object is missing and further processing must be done. To retrieve the correct 3D position of the sphere, two direction vectors (R1 and R2) are cast from the origin to opposite sides of the sphere. The distance (d) between the endpoints of the vectors should be equal to the diameter of the sphere. The length of the rays is incrementally scaled by a factor t until the distance between the two endpoints is equal to the real world diameter of the sphere. The steps are defined using a binary search algorithm. Finally, once the two endpoints are at the correct distance from each other, the midpoint is calculated to find the correct position (P2) of the sphere as can be seen in Figure 4.

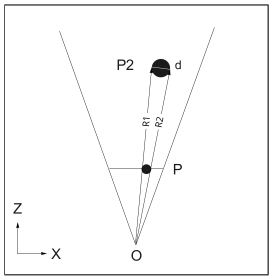


Figure 4: Finding the correct position P2 using sphere diameter d

2.3 Implementation

The method demonstrated in the previous section was implemented using Python. Python has several libraries that make data processing much easier. In this work, Python 3.9 was used alongside the latest compatible distributions of OpenCV (CV) for Python, Pandas (PD), and Numpy (NP). For loading and saving the data-set Python pickle and GZIP were used, however, other libraries can be used based on the data set file format. Each of the different input streams, provided by the Hololens 2, were stored in a Pandas dataframe, a dataframe can be thought of as a table with columns describing each entry. As an index the timestamp of each image frame was used, in this way different input sources with different frame rates can be compared by using the nearest timestamp. Each AHAT and LT image frame can be represented as a 2D array, each entry holding a 16bit value representing the IR intensity value. Some of the OpenCV methods used in this implementation only operate on 8 bit channels as such, the values must first be converted to 8bit values. If the values are normalized directly using OpenCVs normalization function, low IR values will be scaled upwards which would make isolating the markers from the background harder to do. As such binarization is done first followed by reduction of the values to 8bit values.

Binarization

The first step of the process is Binarization, or thresholding. As thresholding can be done on the source image, without normalization, applying a threshold based on the 99.5th percentile would isolate the reflective markers from its surroundings. To avoid the loss of data in areas of interest a blur filter is applied to smoothen out any artifacts around the edges of the sphere, The code below shows how a simple threshold can be performed on an image frame.

```
percentile = NP.percentile(frame, 99.95)
_, frameThresholded = CV.threshold(frame,
    percentile, 255, CV.THRESH_BINARY_INV)
```

The use of `CV.THRESH_BINARY_INV` inverts the input frame such that the highest values are set to the lowest values and vice versa. This is done to allow for blob detection in OpenCV as it detects outliers (blobs) on frames with a white background. The next step of the process is to apply blob detection, but before that is done the image must be reduced to an 8bit image. That can be done by using the following Numpy code: `(frame/256).astype('uint8')`

Blob detection

Now that the markers have been isolated and the frame converted correctly, blob detection can be applied. Blob detection can be implemented with different parameters. Combinations of different parameters were used to find the optimal setup for consistent detection. By setting the minimum area to 25 and circularity value to 0.9 the spheres were detected consistently. Circularity is defined as $4\pi Area/perimeter^2$, with a circle having 1.0 circularity and a square 0.785. With the correct parameters set any remaining artifacts, that were not removed during binarization, were not detected. Figure 5

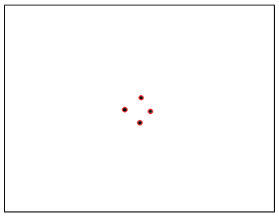


Figure 5: Blobs highlighted on Binarized Frame

visualizes the output of blob detection, keypoints, on the binarized input frame. As output the detection returns the center of each blob as a 2D point and the diameter of the blob. With these results the marker positions can then be retrieved using the final stage of processing.

2D to 3D

The most challenging step of processing is retrieving the 3D positions from the 2D positions. The first step of this process is unprojecting the 2D point. By using the supplied pixel-to-camera space look-up table, the point is unprojected on the plane (x, y , z=1). This means that the point is considered to be very close to the screen. In Figure 6 the result of the unprojected keypoint can be seen as point P. To get to the correct position P2, the direction vector OP must be scaled by a value *t*. Due to the lack of camera properties and the inaccuracy of the HoloLens 2's inbuilt depth sensor, it is harder to calculate *t* accurately. If the camera properties were known the focal length of the camera could be extracted. Commonly used methods [10] for depth detection rely on the focal length of the camera to apply similar triangle ratios using the known diameter of the sphere to retrieve *t*. The proposed solution,

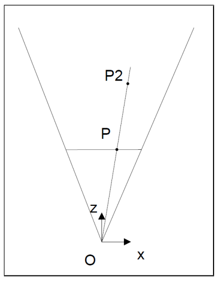


Figure 6: Uprojected 2D point P scaled to correct position P2

in this paper, is based on a binary search approach that also utilizes the known diameter of the sphere. When casting two vectors R1 and R2 from the origin to two opposite sides of the sphere, seen in Figure 4, the distance between the two endpoints, of each vector, should be equal to the diameter of the sphere. While the distance between the 2 points is not equal to the diameter, continue searching for the correct length of the two vectors. To find the starting points of R1 and R2, the pixel positions of the right and left side of the sphere are unprojected. This can be done by using the keypoint output of CV's blob detection, it provides the pixel positions of the sphere's center, and its diameter (size). As a lookup table is used to retrieve the 3D points, the 3D positions are bound to the pixel centers. Linear interpolation can be applied to overcome this limitation providing sub-pixel accuracy, but introducing floating point errors. The code fragment below shows how to retrieve R1 and R2's initial direction vectors for use in searching without linear interpolation:

```
# For every valid blob
for keypoint in validKeypoints:
    # 2D coordinate of the center of the blob
    coord = keypoint.pt
    x = int(coord[0])
    y = int(coord[1])
    # Radius of blob
    r = int(keypoint.size / 2)
    # Unproject right point and normalize to get
        direction vector R2
    coordinatesRight =
        get_lut_projection_pixel(lutAhat, x + r, y)
    coordinatesRight = coordinatesRight /
        np.linalg.norm(coordinatesRight)
    # Unproject right point and normalize to get
        direction vector R1
    coordinatesLeft =
        get_lut_projection_pixel(lutAhat, x - r, y)
    coordinatesLeft = coordinatesLeft /
        np.linalg.norm(coordinatesLeft)
    # Use binary search algorithm
    coordinates3D =
        get_distance_binary(coordinatesRight,
        coordinatesLeft)
    points.append(coordinates3D)
```

Once the direction vectors have been retrieved correctly the code below can be used to find the final position of the sphere:

```
def get_distance_binary(coordinatesRight,
     coordinatesLeft):
    R1 = coordinatesLeft
    R2 = coordinatesRight
    end = 2000.0
    start = 1.0
    # Target diameter
    targetVal = 13.0
    # Current Diameter
    currentVal = -1
    # Test for an epsilon to prevent floating point
        precision errors
    while abs(currentVal - targetVal) > 0.000001:
        mid = (start + end) / 2
        currentVal =
            np.linalg.norm((rightSideVector * mid)
            - (leftSideVector * mid))
        if currentVal > targetVal:
            end = mid
        else:
            start = mid
    # Sphere position P2
    return (rightSideVector * mid + leftSideVector
         * mid)/2
```

The correct position of the sphere is the midpoint of the two endpoints which is returned by `get_distance_binary(coordinatesRight, coordinatesLeft)`. NP is used to perform component-wise addition and subtraction as well as normalization. By implementing these steps the resulting point should be close to the actual position of the reflective marker.

3 Experimental Setup and Results

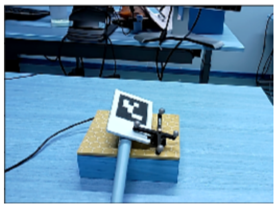
3.1 Setup

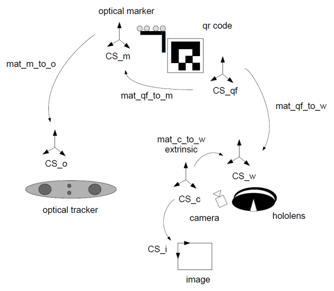
Current state of art the optical tracking systems provide sub-millimeter tracking accuracy, the tracking algorithm presented in this paper does not provide results that are consistently accurate enough to be used independent of other technologies. The accuracy of the AR tracking method was calculated by testing the resulting positions of the algorithm against the results produced from the Polaris Vega VT optical tracker [5]. Each reflective sphere attached to the marker has a diameter of 11.5mm[5], knowing the actual diameter of the sphere allowed for the calculation of the distance of the camera to the marker. In order to test the results, both positions produced from each of the different systems must be represented in the same coordinate system. To do so a reference QR code was used, QR codes are easy to detect by both system. The QR code was used to register the world coordinate system of both the Hololens 2 and the optical tracker, this introduces a small error or noise in the calculation of the true positions. Figure 7a shows the QR code and marker used for evaluation. Figure 7b shows how different positions can be brought into the same coordinate systems, with CS_x representing a coordinate system, mat_a_to_b representing the 4x4 transformation matrix to go from CS_a to CS_b. M represents the optical marker, o the optical tracker, qf for the front of the qr code, w as the world of the Hololens and finally c represents the AHAT camera.

To evaluate the positions of the AR tracking method, both positions are brought to camera space, CS_c. Once in CS_c the evaluation is done by calculating the euclidean distance of the two points for each frame. Since the optical tracker and AHAT cameras are not synchronized, they have different frame rates starting at different times, the frames are compared to the nearest frame within 25 ms. If no frame is found within that time interval it is skipped, as it is not possible to perform an appropriate evaluation. Using a smaller value for the allowed time difference reduced the amount of usable frames, shifting the results due to outliers.

3.2 Results

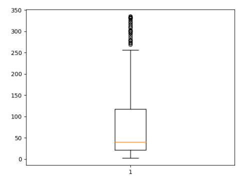
The results presented are based on the assumption that there are no highly reflective objects present in each of the frames


(a) Optical marker with reference QR code used for tracking

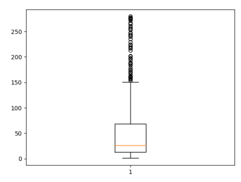


(b) Coordinate system transformations to get true positions using QR code

Figure 7: Coordinate systems transformations using QR code

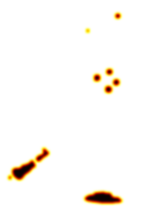


(a) Box-plot of euclidean distance between the true positions and calculated positions using 11.5mm as expected diameter



(b) Box-plot of euclidean distance between the true positions and calculated positions using 13.0mm as expected diameter

Figure 8: Distance results Box-plots


(a) Using the 99th percentile value for binarization threshold leads to artifacts being picked up

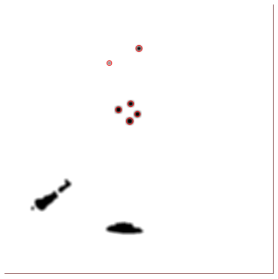

(b) Blob detection on executed on figure 9a picks up unwanted artifacts

Figure 9: Blob detection using a lower threshold value during binarization

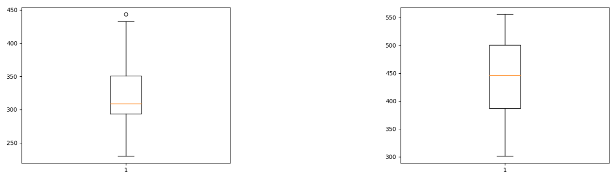
other than the spheres themselves. Moreover, it is presumed that there are 4 optical spheres in each scene and that only 1 sphere needs to be matched correctly. The other positions can be retrieved by performing a fixed set transformations on the initial sphere. As the center of the marker was sometimes picked up due to high intensity readings, the algorithm allows for 1 extra object to be picked up and compared to the true positions. This prevents the spheres from being replaced by the center of the marker, leading to incorrect readings. The extra reading is, assumed to be, implicitly excluded from the evaluation, as it should not be closer to the true positions of the spheres than the calculated values of the spheres themselves. Moreover, results using the depth map of the Hololens were recorded and were found to be measurably worse as such they were not used in the evaluation. The results are evaluated on 1 data-set with approximately 1000 frames. A different data-set may require adjustment of the algorithms parameters.

Based on a set of different parameters the results of the algorithm were recorded. Initially the average distance between the points was calculated and the minimum resulting value was also recorded. Outliers inflated the average value so a box-plot was used instead to provide more information regarding the results. By trying to find the actual diameter of the reflective sphere, the result were less promising than expected. The inter-quartile range (IQR) was considerably large. Figure 8a shows a box-plot of the results found with the box indicating the IQR, the whiskers indicating 1.5 IQR + quartile 3 and 1.5 IQR - quartile 1, the red line indicating the median and the circles indicating outliers.

After some evaluation and testing of different parameters, it became clear that during binarization the size of the sphere became smaller, as the sphere's outer edges had lower intensity values. This added an offset on the initial position of the right and left direction vectors used to calculate the distance. Reducing the binarization threshold would lead to the correct sphere size however, reflective artifacts were detected causing the algorithm to function incorrectly. An example of such behaviour can be seen in Figures 9a and 9b.

To counteract the effects of binarization, an increased expected value can be used to find better results. Figure 8b shows how the results improved significantly when the expected diameter was increased. The minimum distance value found for an expected diameter of 11.5 mm was 2.23mm while the minimum for an expected diameter of 13mm was 1.04mm. Other values for expected diameter were calculated presenting more information with regards to the accuracy of the results. Section 3.3 will show how changing the expected diameter and other parameters influence the results.

Although the minimum value found is almost at the target sub-millimeter accuracy, the median of the results is still too high. At 25mm the results are considered inaccurate. To further examine the effects of the outliers and the most influential factors causing these outliers, each frame with a distance measure higher than the median was stored in an array. The depth value for each of these frames was logged in a box-plot, by doing so it became clear that the depth of the spheres from the Hololens significantly affected the accuracy of the measurements. Figures 10a and 10b show that the median depth for results less than median distance is 320mm while those greater than median value had a depth of 450mm. With the lowest values being 230mm for those less than median distance and 300mm for those greater, and the maximum of each being 450mm and 550mm, respectively. This shows that the algorithm has an effective distance of 400mm with any values above that leading to significantly worse results. This is due to the resolution of the IR cameras on the Hololens. The sphere becomes represented by less and less pixels as it gets farther away, as such the initial positions of the 2 vectors depth detection are inaccurate.



(a) Depth of spheres with distance less than median

(b) Depth of spheres with distance more than median

Figure 10: Depth of spheres above and below median value

3.3 Influential parameters

To overcome the restrictions posed by the resolution of the cameras, sub-pixel accuracy can be achieved by performing linear interpolation during unprojection. Several test were performed using linear interpolation with different expected measures. The result of using linear interpolation with an expected measure of 14.5mm produced a minimum of 0.2mm. This result is promising however the median value increased to 27mm. While using 13mm as expected diameter the result had less accuracy, with 2.03mm as a minimum and 39mm as a median. When using linear interpolation the algorithm approaches a global minimum median of 23mm at approximately 15mm as expected diameter. Without linear interpolation the global minimum median is obtained at approximately 13mm. Statistical analysis, namely the Wilcoxon sign test[11], was performed on the different results to determine the significance of changing parameters. The output of the Wilcoxon test is a p-value. A very small p-value, ¡ 0.05, indicates that the changed variable produces a significant impact on the results and visa-versa. The Wilcoxon test was used on different pairs of parameters, the first of which was a comparison between interpolation and no interpolation on an expected value of 13mm as diameter. The test produced a p-value of 10^{-8} meaning that interpolation has a significant effect on the results produced by the algorithm. using linear interpolation, 14.5 mm, and 15mm expected diameter were used as input to test the significance of changing the expected diameter. The resulting p-value was 10^{-12}, this shows that the expected diameter of the algorithm also produces a significant difference in the results. Its is hard to concluded the optimal parameters for the presented method as only one dataset is provided. The parameters presented in this paper may not work the same for different setup or frames with a different environmental context. As such linear interpolation, expected diameter and the depth of the sphere from the scene should be highly considered when implementing this method.

4 Responsible Research

Surgical navigation systems are used in life threatening situations where accuracy, robustness and reliability are necessary. The algorithm presented in this research must still be implemented for use on the Hololens 2. The main results of this paper show the accuracy of the tracking algorithm, but do not show the robustness and reliability of the algorithm in a real world context. As such the methodology presented here must not be used as a final implementation, but as an experimental basis. The robustness and reliability of the presented method must be tested extensively in a real world context to ensure that it meets medical specification. Once tested, the code can then be ported to the HoloLens 2 for further testing on the machine itself to ensure that it meets performance requirements. After sufficient testing has been performed on the reliability, robustness, and performance to ensure that they meet medical requirements alongside the results presented in this paper relating to accuracy, the ethical concerns of this research would be resolved.

4.1 Reproducibility

The methods presented in this paper are easily reproducible, however they depend on the availability of an augmented reality headset and its utilities. The algorithm presented in Section 2 uses Python as well as publicly available Python libraries. Each of these libraries has extensive documentation making their operation trivial once the processing stages have been defined. The headset manufacture must provide the camera properties, intrinsic and extrinsic, or alternative utilities such as a pixel-to-camera space lookup matrix to allow for the correct operation of the presented methods. The Hololens 2 comes supplied with the required utilities and exposes the camera data for research use, as such reproducing these methods for the HoloLens 2 should not be challenging. The source code can be found at [12], along with the data-set.

5 Discussion

The works of Kunz et al.[8] have shown significant results in AR marker based tracking, however the setup used was controlled relative to that of this paper. As such a consistent accuracy of 0.76mm is hard to achieve in real world environments with no fixed depth for the markers. The method used in this work tries to overcome some of the limitation presented by a real world setup. In real world environments, however, different factors such as ambient IR radiation and rapid movements influence the correct operation of the discussed tracking method. Nonetheless, the inaccuracies of current IR based tracking methods, in practical environments, are still considerable. In order to achieve better, more accurate, results the Hololens should be fitted with an improved depth sensor to reduce the influence of external factors. By using the true depth, emulating a better depth sensor, to unproject the 2D points found during blob detection, the target sub-millimeter accuracy was achieved consistently. Other methods that do not rely on the IR sensor accuracy or use the methods presented as a basis, might be better approaches for improving accuracy. Examples of such methods are using the stereo greyscale cameras and deep learning techniques to identify the points in a greyscale image, Even then, the induced performance overhead might not be suitable for the HoloLens 2. As such there is still room for growth in the field of AR surgical navigation, due to the limitations of current AR technology.

5.1 Future work

The method presented in this paper can still be improved. Deeper examination of frames with high distance errors shows that the expected distance of the sphere must be adapted based on the reflectivity of the spheres and the depth of the spheres. Different calculation can be done to adapt the expected measure based on the pixel size of the sphere on the screen. As the marker goes farther away from the screen, less pixels represent each sphere. The lookup table allows for the unprojection of 2D points at a specific pixel position. To overcome this limitation linear interpolation is done on the unprojected pixel positions. This introduces another error factor that can be removed with the use of higher resolution frames. Some tests were performed using a combination of the method mentioned in this paper with the depth map of the HoloLens 2. When the presented method failed the depth sensor provided a better results. However it is hard to know when the results of the depth sensor can be depended on. Using the ground truth to evaluate when to use the depth sensor showed better results. It reduced the median to 20mm and obtained a minimum value of 0.2mm.

5.2 Limitations

The main limitation of this research is that the optical tracker's ground truth values must be computed in the Hololens world coordinate system. A reference QR code is used to register the points, however an error in the calculation of the QR code positions exists as both systems are not absolutely accurate. Moreover, floating point errors add up during the transformation from one coordinate system to another, increasing the error in the ground truth calculations. The frames rates of both sources are not synchronized, which means that the target may have moved within the allowed time difference. Another considerable limitation of this research is the resolution of the IR frames, with higher resolution frames the edges of the spheres can be identified with ease and mapped to the correct pixel positions. This would improve the initial positions of the 2 vectors used to calculate the size of the sphere. By finding a better way to localize the true positions of the spheres the error in calculations can be reduced, and the required sub-millimeter accuracy may be achieved as commercially available AR technologies improve.

6 Conclusion

Surgical navigation systems are foundational to less invasive surgical procedures. As current systems rely on stationary screens and trackers they introduce impediments on a surgeons focus. AR SNSs might be a great alternative to the use of fixed equipment. Nevertheless, given the current state of the Hololens 2 its cameras and sensors are still not accurate enough to produce consistent results using IR marker based tracking. The method presented in this research attempts to operate on real world conditions where random movements were used and an accurate depth map was not present for each frame. This produced a minimum distance error value of 1.04mm. Although this value is almost at sub-millimeter accuracy it is not consistent. The median distance error presented by this method was approximately 25mm which is too high for use in medical applications. With the introduction of more advanced AR headsets, AR SNSs will become more common in medical applications. However, the ethical implications of using AR headsets for surgical navigation must be addressed before wide spread use.

References

1. Medtronic. *Surgical Navigation Systems - stealthstation* June 2020. https://www.medtronic.com/us-en/healthcare-professionals/products/neurological/surgical-navigation-systems/stealthstation.html.
2. School, M. M. *Surgical navigation* Sept. 2021. https://med.uth.edu/orl/texas-sinus-institute/services/surgical-navigation/.
3. Mezger, U., Jendrewski, C. & Bartels, M. Navigation in surgery. *Langenbeck's archives of surgery* **398,** 501–514 (2013).
4. Ambrosini, P., Thabit, A. & Benmahdjoub, M. *HoloNav: HoloLens as a surgical navigation system* 2022.
5. *Vega / vicra optical measurement tools and accessories* Mar. 2022. https://www.ndigital.com/optical-measurement-technology/polaris-tools-and-accessories/.
6. Benmahdjoub, M., van Walsum, T., van Twisk, P. & Wolvius, E. Augmented reality in craniomaxillofacial surgery: added value and proposed recommendations through a systematic review of the literature. *International Journal of Oral and Maxillofacial Surgery* **50,** 969–978 (2021).
7. Gsaxner, C., Li, J., Pepe, A., Schmalstieg, D. & Egger, J. *Inside-Out Instrument Tracking for Surgical Navigation in Augmented Reality* in (Dec. 2021).
8. Kunz, C. *et al.* Infrared marker tracking with the HoloLens for neurosurgical interventions. *Current Directions in Biomedical Engineering* **6,** 20200027 (Sept. 2020).
9. Vtieto. *Hololens research mode - mixed reality* Apr. 2022. https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/research-mode.
10. Rosebrock, A. *Find distance from camera to object using python and opencv* July 2021. https://pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/.
11. solutionsstatistics. *The Wilcoxon sign test in SPSS* June 2021. https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/the-wilcoxon-sign-test-in-spss/.
12. Ambrosini, P. *Holonav Repo* 2022. https://gitlab.tudelft.nl/pambrosini/holonav.