AR Phone Way-finding Using Imagery and Navigation for Indoor Environments

Supervised by: Dr Henry Gardner
Given by: Yukun Hao
U6013736
09/06/2020
Brief Introduction

1. This project extends Zhibo Zhang’s AR navigation software, to improve its usability, accuracy and robustness for indoor navigation.

2. New AR indoor navigation software has been built, with a QR Code-based re-localization function.

3. Programming Requirements: C#, Unity3D

4. Other skills required:
   • Socket-server programming;
   • Heuristic evaluation;
   • Image processing techniques

5. Why this project?
   • Indoor AR navigation using QR codes depends on the angle that you are viewing these codes.
   • This project corrects for the angle of view of QR codes using image processing techniques.
   • It also synchronizes indoor navigation automatically using QR codes rather than requiring manual synchronization.
   • Explained more in the following slides....
Background: AR

Augmented Reality – Mixing real-world environment with virtual objects.

Image From: https://imgix.bustle.com/mic/eelbogf5seevwdxnrgrichzmcj2o4m8stl8g6sffa5qscg4lqbdx9uhtfox7lbpf.jpg
Background: QR Code

Quick Response Code – a flexible way of storing information

Image From: https://coolshell.cn/wp-content/uploads/2013/10/QR-Code-Overview-768x461.jpeg
Background: Indoor Navigation

Intuitive to use AR for indoor navigation
Background: Indoor Navigation

GPS: Inaccurate

1. In indoor environments, GPS location information is not sufficiently accurate for navigation.
   • Navigation is extremely sensitive to even very small location errors (several meters).

2. Even if it were completely accurate, GPS cannot recognize which floor you are in if your building has many floors.
Background: Zhang’s Software

To overcome drawbacks of GPS, Zhang allowed users to manually re-localize themselves using known “synchronize points”
What’s the main problem?

Users should manually re-locate themselves by pressing buttons.
Users should manually decide their direction.

Inconvenient and depends on users behaviors!
Methodology: Introduction

1. Instead of using Synchronize Point, I use QR Codes that are mounted on surfaces in a building.

2. My app takes a photo of a QR Code, and send it to a server

3. The server decodes the QR Code, and calculate a relative orientation that the photo was taken from.

4. The location and orientation information is sent back to the client and the client then displays the correct navigation direction.
Methodology: Using QR Code

Use paper with QR Code instead of Synchronize Point
Methodology: Take a photo

Take a photo, decode the QR Code to know the location information.
Methodology: Decide Orientation

A relative orientation

If we can know:
1. The relative orientation between the camera and the QR Code on the wall
2. The orientation of the QR Code

Then we can know:
Users orientation in the map model

\[ \text{Orientation}_{\text{user}} = \text{Orientation}_{\text{relative}} + \text{Orientation}_{\text{QR Code}} \]
Methodology: Decide Orientation

Calculate the relative orientation based on the photo

If the relative degree is large, the left edge would be shorter or longer than the right edge of the QR Code.

Key principle:
Thing looks bigger if closer, and smaller on the contrary.

I.e. taking a photo from the right side:
\[ \text{Length}(\text{Edge}_{\text{right}}) > \text{Length}(\text{Edge}_{\text{left}}) \]
Methodology: Setup a Server

A server has been setup, to:

1. Decode QR Code in a photo being received.
2. Calculate the relative orientation.
3. Send the location and orientation information back to the client.

Why need a server:

1. Separates the navigation function and re-localization function.
2. Provide additional information by enabling “live QR Codes”.
3. People can keep adding and replacing QR codes on the server without updating the client.
4. Allows this project to be extended further in the future.
Results and Evaluation: Experimental setup

The server configuration

<table>
<thead>
<tr>
<th>Hardware/Software</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer</td>
<td>Lenovo ThinkPad E580</td>
</tr>
<tr>
<td>CPU</td>
<td>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</td>
</tr>
<tr>
<td>Main memory</td>
<td>8 GB – DDR4 2400MHz</td>
</tr>
<tr>
<td>OS</td>
<td>Windows 10 x64</td>
</tr>
<tr>
<td>GPU</td>
<td>Radeon (TM) RX 550 (2GB)</td>
</tr>
<tr>
<td>Hard disk</td>
<td>KBG30ZMT 128G TOSHIBA</td>
</tr>
<tr>
<td>NIC</td>
<td>Realtek RTL8168 Gigabit Ethernet Controller</td>
</tr>
</tbody>
</table>

The client (phone) configuration

<table>
<thead>
<tr>
<th>Phone configuration</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phone</td>
<td>Google Pixel 3 XL</td>
</tr>
<tr>
<td>OS</td>
<td>Android 10</td>
</tr>
<tr>
<td>Google Play Services for AR</td>
<td>Google Play Services for AR unlocks augmented reality (AR) experiences built using ARCore.</td>
</tr>
<tr>
<td>CPU</td>
<td>Snapdragon 845</td>
</tr>
<tr>
<td>Storage</td>
<td>128GB</td>
</tr>
<tr>
<td>Rear camera</td>
<td>12.2 MP</td>
</tr>
<tr>
<td>Screen size</td>
<td>6.3 inch</td>
</tr>
<tr>
<td>Resolution</td>
<td>1440 * 2960</td>
</tr>
<tr>
<td>RAM</td>
<td>4 GB</td>
</tr>
</tbody>
</table>

Note: The Hanna Neumann Building was closed due to the COVID-19 so we were unable to use the same navigation information as in Zhang’s project.

Thus, the software was tested in another building, and the test only focused on the QR Code recognition and orientation calculation functions.
Results and Evaluation: Time Performance

<table>
<thead>
<tr>
<th>Process</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total time</td>
<td>3101</td>
</tr>
<tr>
<td>Send picture to server</td>
<td>1713</td>
</tr>
<tr>
<td>Decode QR Code</td>
<td>136</td>
</tr>
<tr>
<td>Calculate relative orientation</td>
<td>14</td>
</tr>
<tr>
<td>Receive position information</td>
<td>395</td>
</tr>
<tr>
<td>Reset position in client</td>
<td>742</td>
</tr>
<tr>
<td>Others</td>
<td>23.93%</td>
</tr>
</tbody>
</table>
Results and Evaluation: Orientation Accuracy Performance
Results and Evaluation: Heuristic Evaluation

Based on heuristic evaluation principles includes:


Problems found:

1. Some useless buttons and text;
2. No guidelines;
3. Toggles not easy to be understood;
4. Insufficient error management;
5. Insufficient feedback;
6. Button color and text color;
Conclusion and Future Works

I have implemented an automatic re-localization function to extend Zhang’s software to improve its use for indoor navigation.

But still, there are some limitations of my implementations:
- Distance of taking the photo,
- Orientation calculation affected by the distance,
- Long waiting time,
- Other usability issues can be improved.

The re-localization function can be improved in the future, possibly with deep learning model and computer vision techniques.
Acknowledgements

Dr Henry Gardner

Mr Zhibo Zhang

Miss Yujiao Shi
Thanks for your time!

Yukun Hao