Virtual Displays

Automatic Region Detection and Rendering for Augmented Reality Interactivity

Alex Haak

supervised by Pedro Sander

Introduction

Augmented reality is the confluence of computer vision and computer graphics. What does that mean? It is the process of detecting objects in real world environments and rendering some virtual content onto it.

Virtual Displays is an application utilizing this, displaying dynamic information seamlessly, woven into your perceptions of reality.

The core functionality is split into two main areas:

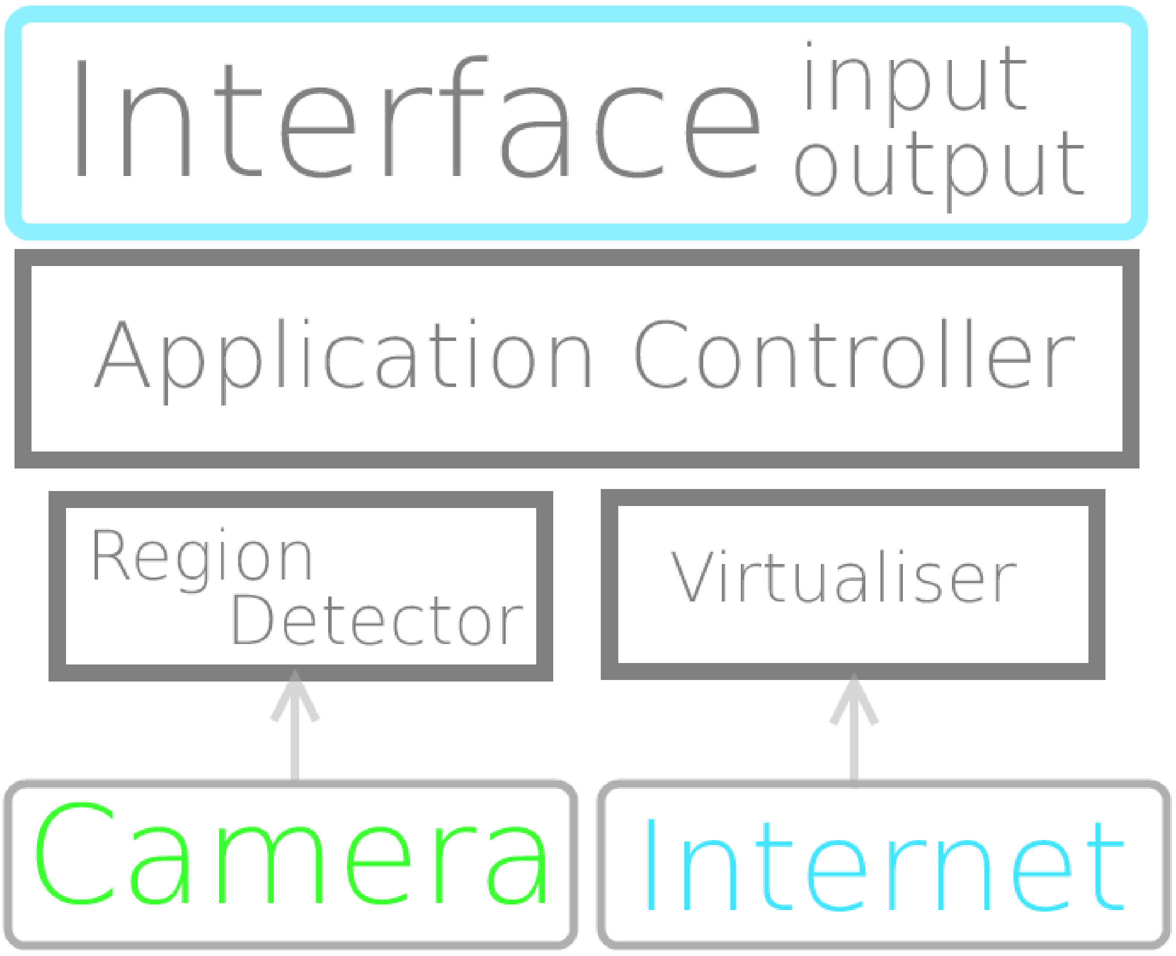
1. Region Analysis – simply the process by which regions are identified.
2. Virtual Content Rendering – using the identified regions, some content can be rendered onto the image.

Features

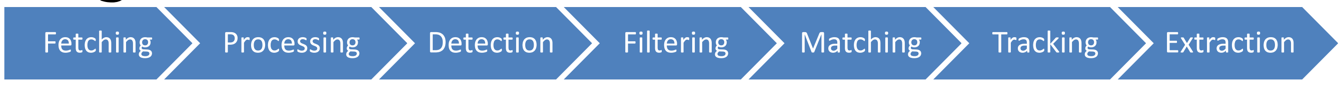
- ✓ Region detection
  - Camera
  - File
- ✓ Display control
  - Source selection
  - Rendering control
- ✓ Virtual content rendering
  - Images
  - Webpages

Architecture

The system is managed and controlled by one main controller. This interacts with both core components each frame as well as working with the Interface component to interact with the user.

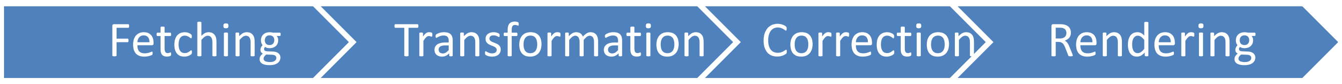


Region Detector



The region detector is responsible for region analysis. The process for region detection as outline above, starts by fetching content from a camera.
This input is then processed with appropriate algorithms that make detection possible. Contours are then detected in the input. These are filtered down and matched with stored regions.
Tracking is used to anticipate where contours will be and finally extraction uses a perspective transformation and image processing to extract the content from the region.

Virtualiser



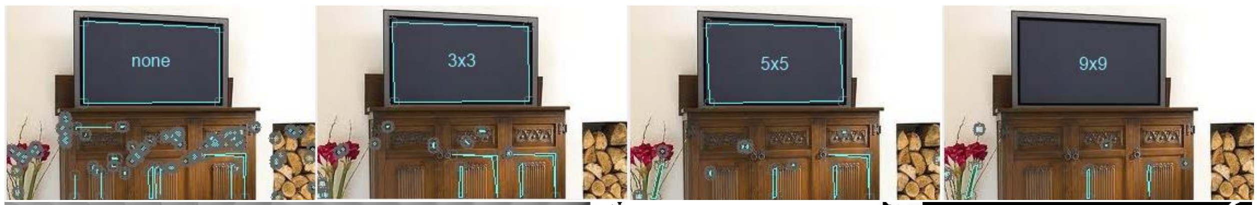
The Virtualiser component is responsible for rendering the virtual content over the input. The process starts by fetching the virtual content from either the internet or the user's file system.
A perspective transformation is applied to the content so that it matches the perspective of the perceived region.
Some content correction also takes place before rendering so that the virtual content fits aesthetically with the environment.

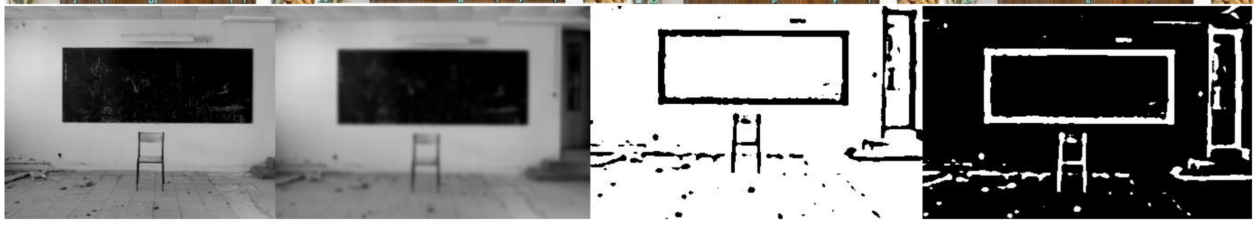
Measuring Success

1. Firstly the number of UPS (updates per second) and FPS (frames per second) are an indication of program updates.
   Quicker = better
2. Secondly a measure of the success and failure rate for detection algorithms. Failure rate includes false positive and true negative

Implementation

- Using Java with OpenCV for computer vision functionality
- HTML rendering was completed with html2Image.
- Application was deployed to the Android platform using Android SDK.





Conclusion

The aim of the project was to use object detection and computer graphics to display dynamic data. This was achieved well technically with test cases largely satisfied. However, an appreciation for user experience was absent during the project. The application made good use of various object detection methods to best detect the regions. The resulting program has also efficient enough for real-time process with an average of 16 UPS.