



COMP0073 – MSc Computer Science Project

Investigation of new Smart Space Services Using Passive Observer Technology

Deployment and User Manual of the Proof of Concept Deliverable

Constantin Ulbrich
Student Number: 15010034

Internal Supervisor: Prof. Dr. Yuzuko Nakamura

External Supervisor: Fergus Kidd

Degree: MSc Computer Science

Submission Date: September 11, 2020

This report is submitted as part requirement for the MSc Computer Science degree at UCL. It is substantially the result of our own work except where explicitly indicated in the text. The report will be distributed to the internal and external examiners, but thereafter may not be copied or distributed except with permission from the author.

Department of Computer Science
University College London

Deployment Manuel

The SmartVision proof of concept system is a locally hosted application. Consequently, it requires some initial setup steps to run the application on any machine.

SmartVision System Setup (Creation of the local environment)

1. Download the project's zip file containing the source code of the SmartVision System in the file COMP0073_SmartVision_Prototype.
2. Open the folder COMP_SmartVision_Project as your working environment in an editor of your choice (this project was implemented in Visual Studio Code).
3. Install and host a MySQL database with the database manager phpMyAdmin locally. Download the WampServer (from <https://www.wampserver.com/en/>) or a MampServer/LampServer for IOS and Linux machines respectively. The windows download includes the MySQL database and phpMyAdmin database manager. For the other operating systems, one might has to download phpMyAdmin extra (<https://www.phpmyadmin.net/downloads/>).
4. Open phpMyAdmin, navigate to Import, and upload the database script "SmartVision_Database_Setup_02082020.sql". This script can be found in the folder SmartVision_Database.
5. The SmartVision System comes with a virtual environment. Navigate to SmartVision_Env, Scripts, and then to python.exe. Select the python interpreter as the projects default interpreter in your code editor for the SmartVision application. This should allow to immediately have a working Python interpreter with all the required packages installed. In case this should not function, one has to set up the environment manually. The system requires a Python interpreter (Python 3.8+) with the following non-standard packages installed:

- ❖ opencv-python 4.4.0.42
- ❖ azure-cognitiveservices-vision-computervision 0.6.0
- ❖ azure-cognitiveservices-vision-customvision 3.0.0
- ❖ azure-cognitiveservices-vision-face 0.4.1
- ❖ azure-storage-blob 12.4.0
- ❖ Pillow (Pillow 7.2.0), a fork of the Python Image Library (PIL)
- ❖ mysql-connector-python
- ❖ unittest
- ❖ matplotlib
- ❖ numpy
- ❖ Flask
- ❖ Schedule

Alternatively, the included *requirements.txt* file contains all the listed libraries. By using the command "pip install -r requirements.txt", all libraries can be installed at once.

Account Information and Configuration of the SmartVision System

Note, the following account information are also available in the *config.py* file.

- ❖ Local MySQL Database:
 - Database Host: localhost
 - Database User: smartVisionUser
 - Database Password: 12345SmartVision
 - Database Name: smartvisiondatabase
- ❖ Azure Computer Vision Service:
 - Endpoint: <https://smartvisioncv.cognitiveservices.azure.com/>
 - Subscription Key: cde849d4eb85496484b33a25bcd1419e
- ❖ Azure Custom Vision Service:
 - Endpoint: <https://uksouth.api.cognitive.microsoft.com>
 - Subscription/Prediction Key: cb53f33c4d774df08d5540a716fb0681
 - Project ID: c690e4e5-6ba8-45be-96a1-c9e6f45552e6
 - Published Project Name: SmartVision_WorkstationObjects_Iteration1
- ❖ Azure Face Service:
 - Endpoint: <https://smartvisionfd.cognitiveservices.azure.com/>
 - Subscription Key: 69e6d60dbec24f6a905c9bead4f6572c
- ❖ Azure Blob Storage:
 - Blob Storage Key:
M96Mp7PEKRtp5tDWTIC1DVkSWib23HeQpYY+7Z9hEWMl5BzHGHN9IiH0QN/8A
HBkc5+yFnfZIW1S2peje9FSfw==
 - Blob Storage Connection String:
DefaultEndpointsProtocol=https;AccountName=frameblobstorage;AccountKey=M96Mp7
PEKRtp5tDWTIC1DVkSWib23HeQpYY+7Z9hEWMl5BzHGHN9IiH0QN/8AHBkc5+y
FnfZIW1S2peje9FSfw==;EndpointSuffix=core.windows.net

System File Structure

A comparted file structure was implemented to fulfil the requirement of a modular design. The folder COMP0073_SmartVision_Project contains the SmartVision system and the Azure Kinect Controller system. All key components of the SmartVision system are in the file COMP0073_SmartVision_Prototype. Below, a description of the pillar files:

- ❖ *SmartVision_Detection_Algorithm*: This file contains all the business logic related to analysing a frame, generating relevant graphs from the statistics stored in the database, handling the up-and download of the captured frames to the blob storage, and post processing the latest analysed frame for the display on the web-interface.
- ❖ *SmartVision_Database*: The folder contains all database logic and queries to connect and communicate with the MySQL database.
- ❖ *SmartVision_Flask*: This folder contains the sub-folders “static” and “templates”, as well as the “server.py” file. The static folder contains any static content, such as images, which should be embedded in the website. The templates folder contains all HTML files, which make up the front-end. The ‘server.py’ file is the heart of the flask application and contains all the backend

logic of the web-interface. This file also connects to the files located in the SmartVision_Detection_Algorithm folder to access the functions of the respective scripts.

- ❖ *SmartVision_Env*: This folder contains the Python virtual environment. This has been created for best practice development standards in order to store and install all relevant packages and libraries, required by the system, into the SmartVision Environment. This should simplify the handover of the project to Avanade's developer team.
- ❖ *Images*: This folder contains temporarily all the images captured by the visual sensor, before those are further processed by the SmartVision algorithm.

User Manual

The SmartVision interface serves the developer for a convenient control and configuration of the SmartVision algorithm. Additionally, the interface displays the data collected by the SmartVision Algorithm in different graphical formats. The user manual explains how the functions can be used:

Running the SmartVision Proof of Concept System

1. If a Azure Kinect is available, it is recommended to use it. Plug in the sensor into one of the USB ports of the local machine and connect it to a power source. In the config.py file, configure to use the Azure Kinect Controller System, as the visual sensor. (If no Azure Kinect is available or it is preferred to use the webcam system, then set “*visual_sensor = webcam*”. Next decide, whether the local machine’s webcam should be used, or the Kinect DK. Equate “*webcam = 0*”, to use the integrated webcam in the machine. To use the Kinect DK, set “*webcam*” to the highest external device number (usually to 1, if the local machine only has one integrated camera).
2. Open a new terminal. Then, navigate to the directory SmartVision_Flask in the working directory COMP0073_SmartVision_Project.
3. Set the Flask Application by executing the command: *set FLASK_APP=server.py*
4. Set the Flask Environment by executing the command: *set FLASK_ENV=development*
5. Run the SmartVision Interface (Flask application) with the command: *flask run*
6. Open the SmartVision interface in your browser: <http://127.0.0.1:5000/login>

Login into the SmartVision Interface:

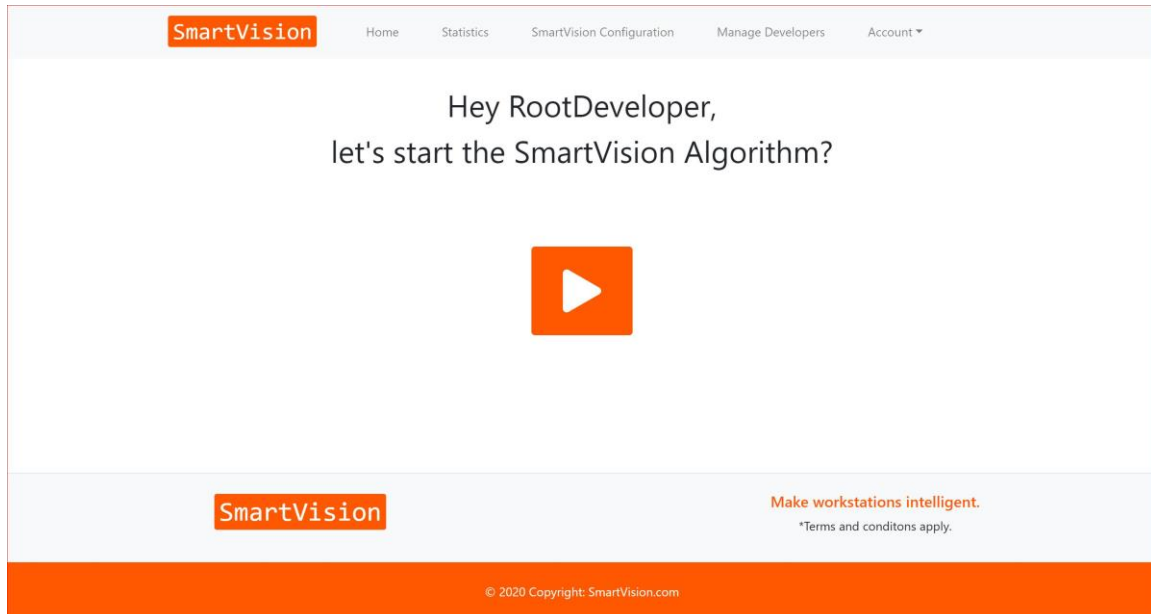
Only SmartVision developers can add new developer accounts to the interface. If you don’t have an account yet, you can use the Root Developer account (username: root.developer@smartvision.com, password: SmartVision12345). After entering the details, press the login button to enter the web-interface.



The image shows a login interface for the SmartVision Prototype. It features a white background with an orange border. The text "Welcome to the SmartVision Prototype Interface" is centered at the top, with "SmartVision" in a blue box. Below this is a light blue input field containing the username "root.developer@smartvision.com" and a masked password "*****". An orange "Login" button is positioned at the bottom center.

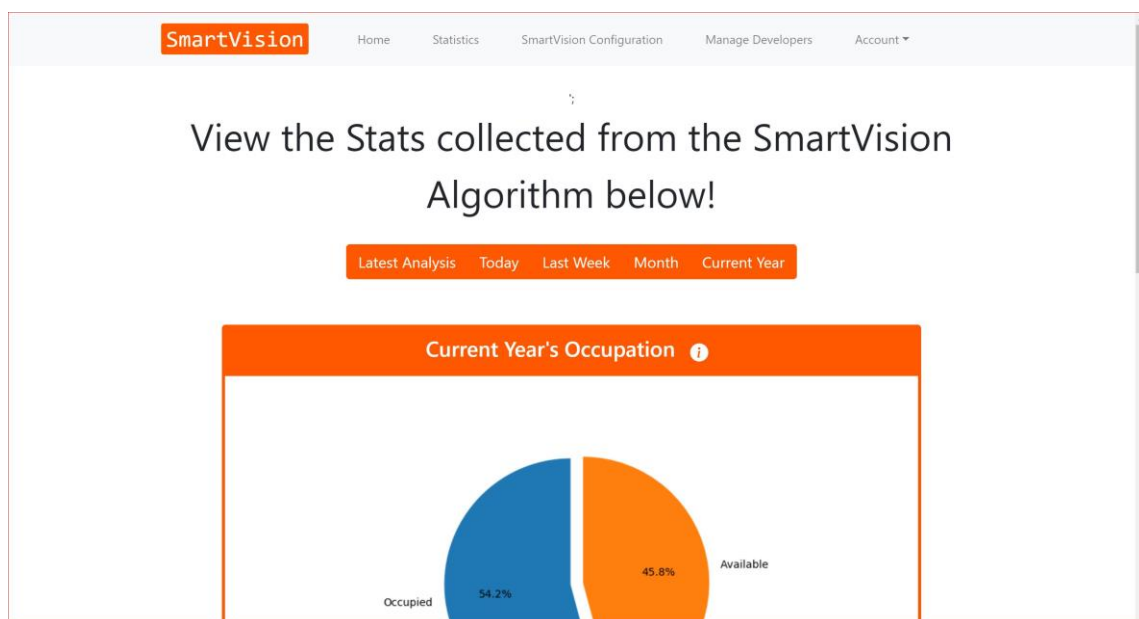
Starting and Stopping the SmartVision Algorithm:

On the Homepage you can press on the play button. This starts the SmartVision algorithm. Whilst the algorithm is running, you can perform other tasks. Press the Stop button on the Homepage to terminate the SmartVision algorithm.



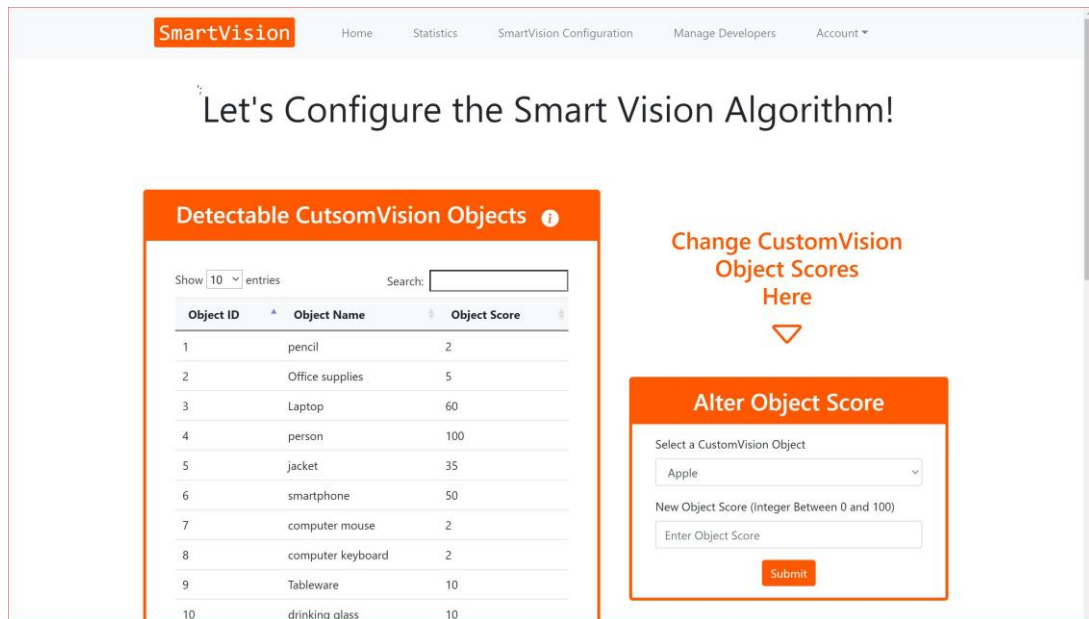
View Statistics Generated by the SmartVision Algorithm:

Press on the Statistics tab in the navigation bar. It prompts you to the Statistics page. Here you can choose a time period for which you want a statistic to be displayed. Note, "Latest Analysis" shows you the information of the last frame, which was analysed by the SmartVision Algorithm.



Configure the SmartVision Algorithm:

A developer can configure the SmartVision algorithms in two ways. The object scores of the objects, which the algorithm attempts to detect in order to determine the occupation of a workstation, can be adjusted. Also, the algorithm's detection sensitivity for the different analysis categories can be altered by changing their probability thresholds. The corresponding tables show the current information and next to the tables a developer can alter the configuration according to her/his needs.



SmartVision Home Statistics SmartVision Configuration Manage Developers Account ▾

Let's Configure the Smart Vision Algorithm!

Detectable CutsomVision Objects

Show 10 entries Search:

Object ID	Object Name	Object Score
1	pencil	2
2	Office supplies	5
3	Laptop	60
4	person	100
5	jacket	35
6	smartphone	50
7	computer mouse	2
8	computer keyboard	2
9	Tableware	10
10	drinking glass	10

Change CustomVision Object Scores Here

Alter Object Score

Select a CustomVision Object

Apple ▾

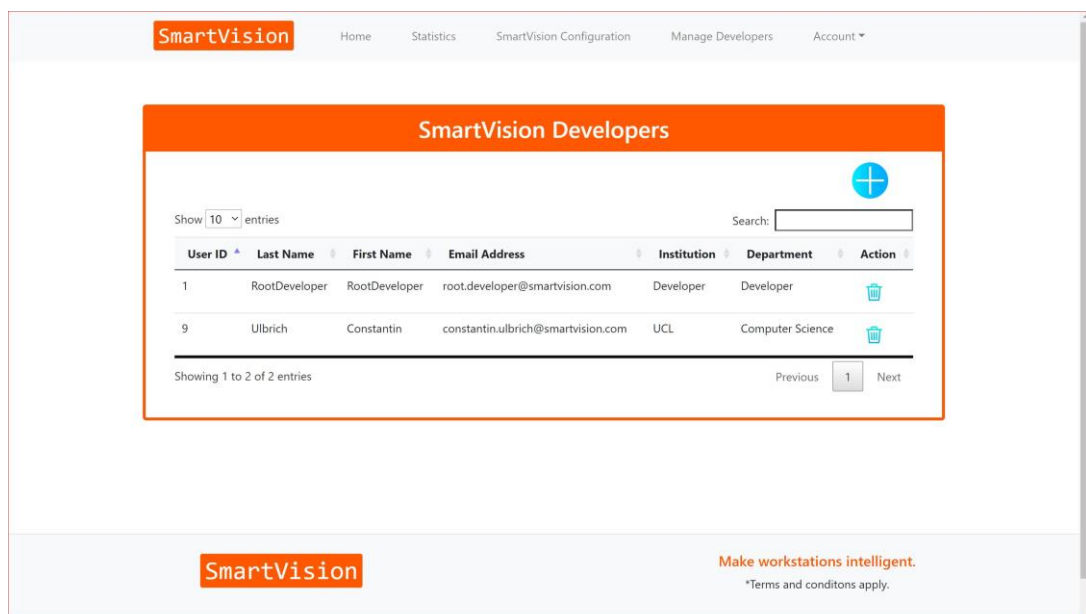
New Object Score (Integer Between 0 and 100)

Enter Object Score

Submit

Manage Developer Accounts:





Existing developers can add new developer accounts or delete existing ones, by pressing at the plus or the trash on the Manage Developers page respectively.



SmartVision Home Statistics SmartVision Configuration Manage Developers Account ▾

SmartVision Developers

Show 10 entries Search:

User ID	Last Name	First Name	Email Address	Institution	Department	Action
1	RootDeveloper	RootDeveloper	root.developer@smartvision.com	Developer	Developer	 
9	Ulbrich	Constantin	constantin.ulbrich@smartvision.com	UCL	Computer Science	 

Showing 1 to 2 of 2 entries Previous 1 Next

SmartVision Make workstations intelligent.
*Terms and conditons apply.

Manage Personal Account:

A developer can change the details of her/his account or the password by clicking “Account” in the navigation bar. A drop-down menu appears from which the developer can select to change the account details or the password.

The screenshot displays the SmartVision web application interface. The top navigation bar includes the SmartVision logo and links for Home, Statistics, SmartVision Configuration, Manage Developers, and Account. The Account dropdown menu is open, showing options for Edit Profile Data and Change Password. The 'Edit Profile Data' form is visible, containing input fields for First Name, Last Name, Email, Name of Institution, and Name of Department, along with a Submit button.

SmartVision Home Statistics SmartVision Configuration Manage Developers Account ▼

Your Account

Edit Your SmartVision Profile

Edit Profile Data

First Name
RootDeveloper

Last Name
RootDeveloper

Email
root.developer@smartvision.com

Name of Institution
Developer

Name of Department
Developer

Submit