

# M30242 Graphics and Computer Vision 2021-22

## Coursework

This assessment accounts for 50% of the overall assessment of the module. It consists of two tasks that assess the learning outcomes 3 and 4.

This assignment document contains two appendices.

### Warnings:

1. The work you submit must be of your own. You may reference the numerous examples on the web, but you must NOT copy any of the examples. Failure to comply with this requirement may result in plagiarism procedures being taken.
2. The programs you submitted must be compatible with the software and hardware settings on standard University lab PCs without requiring installation of any additional libraries or updates. It is your responsibility to make sure all the files are intact and compatible with version of the software provided by the IT Service of the university (see <https://servicedesk.port.ac.uk/> for detailed information)

## Assignment Project Exam Help

### Submission

1. Submit all the deliverables of both Task 1 and 2 in the form of *a single zip file named with your student number* through the *Coursework Submission* folder on the unit base on Moodle.
2. Submission on Moodle will open a week before the deadline and close at 14:59 on the deadline day.
3. The submission must be anonymous. Do **NOT** write your name on any artefact.
4. Emailed work will **NOT** be accepted.

## Task One (50%)

### Application Scenario and Conditions

To control the traffic at the entrance of a narrow tunnel, a computer vision system is used to intercept oversized or speeding vehicles. Any vehicle that exceeds 2.5m in width and/or 30 miles per hour in speed will be diverted or stopped by traffic lights or police officers upon receiving the warnings from the system. Fire engines (assuming that they are mainly red in colour and have a width/length ratio of approximately 1:3) are the only vehicles exempted from the control.

The system consists of a video camera fixed right above the centre of the lane of a straight road and is 7m off the ground. Its optical axis is 30 degrees off the horizon and pointing to the lane along the direction of traffic. A diagram showing the camera configuration is given in the Appendix 1. The resolution of the sensor of the camera is 640X480 pixels. For simplicity, it is assumed that the pixels of the camera sensor are square and each sensor pixel is equivalent to 0.042 degrees in view angle. The camera grabs a video frame (an image) of the lane at 0.1s intervals. It is further assumed that each frame contains only one vehicle. The output frames from the camera will be the only input to the vision system. You should not make any assumptions about the identity of the vehicle contained in a frame before processing the frame.

### Requirements

1. *Detection Functionalities.* Design and implement in Matlab a computer vision application that fully meets the specification. Your application should contain the following basic functionalities: Size detection, Speed detection, Colour detection.

2. *Outputs and Information.* Your program should output the results of the key processing steps, for example smoothed images, colour blobs, centres, outlines or bounding box, and document them in the report (see Requirement 4) in the form of screenshots.

Your program must output the following values or information in the command line window:

```
Car width:      meters
Car length:     meters
Car width/length ratio:
Car colour: Is the car red? Y/N
Car speed:      mph
Car is speeding (Y/N):
Car is oversized (Y/N)
Car is fire engine (Y/N):
```

3. *Test and Verification.* Include the following script (name it as **Test.m**) in your application and use it to test your application. The coursework assessors will run the script to verify the intermediate and final outputs of the application.

```
clear,
close all,
img1=imread("xxx.jpg"),
img2=imread("xxx.jpg"),
my_application(img1,img2)
```

Speeding detection could be tested using image pairs *001.jpg* vs *002.jpg*, *001.jpg* vs *003.jpg*, *001.jpg* vs *004.jpg*, and so on by assuming that the images are taken at 0.1s interval in each case. Fire engine and oversized vehicle should be tested using the relevant images.

4. *Documentation and Report*. Prepare a report of no more than 1000 words that documents the system design, testing and evaluation. The report should

- include discussions of the necessary conditions of and/or assumptions about the application that may help to justify your design decisions. For example, you may assume there is only one car in an image as you could track a car across the frames, but you should not assume that you know the colour of a car or the car stays right in the middle of a road lane;
- provide a concise flow diagram of the entire system to show the important processing steps;
- justify each processing step by discussing or showing the features you want to extract, the method used for extracting the features, and the screenshots of the input and output of the processing step;
- document your test against the application scenarios by providing the inputs, e.g., the names of image files, and outputs, e.g., the screenshot of the speed/width/colour of a vehicle and any command line messages.

Assignment Project Exam Help

#### **Deliverables**

1. Program code and supporting files. The program code should be in M-file format and suitably commented and with instructions to use/run it.
2. A copy of the coursework report that meets *Requirement 4*.

WeChat: cstutorcs

## Task Two (50%)

**Warning:** For this task, you are NOT allowed to use any WebGL library that has built-in functions for creating, drawing, texturing geometric primitives such as spheres, cubes, and so on. You need to generate the vertex data of such objects, i.e., vertex coordinates, texture coordinates and normals, and perform the relevant operations, such as texture mapping, lighting and shading calculations, by yourself.

### Specification

In this task, you are required to create an animation. The scene consists of the planet earth in the middle with a satellite moving around it along a circular orbit within the horizontal plane. The scene is illuminated from top-right by a *directional* light that is at a 60-degree angle with the horizontal plane if viewed in front view.

The earth model is a sphere of radius 20 or larger and mapped with an earth image. The earth rotates slowly around its own vertical axis. An image of the earth is provided for texturing the sphere.

The satellite consists of a main body of a *cube* of size of 2x2x2 and two “solar panels” that are attached to the two opposite sides of the main body through two connection “rods”. The cube is *golden* in colour except for the sides where the solar panels are attached which are *dark grey*. The connection rods are *cylinders* of 0.2 in diameter and 0.7 in length. The solar panels are *blueish* thin rectangular objects of 2.0x5.0 in size. For simplicity, we assume that the panels always face upwards. A *golden* antenna *dish* of a diameter of 4.0 is attached to one side of the satellite by a *cylinder* rod of 0.3 in diameter and 1.0 in length. The antenna dish will constantly face the earth while orbiting. All the connection rods are *light grey* in colour (for the shape of the satellite see illustration in Appendix 1).

The animation will be interactive: You should be able to control the radius of the circular orbit (with the left and right arrow keys) and the speed (up and down arrow keys) of the satellite at runtime. You should have full viewport/scene-navigation control: translations along x- (shift plus mouse drag), y- (alt plus mouse drag) and z-direction (mouse wheel) and rotations around x- and y-axes (mouse drags). The translation controls should be independent of the rotation controls.

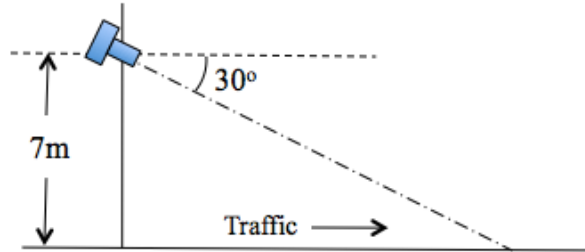
Your application should work with standard browsers on the PCs in the University Labs without requiring any special set-up or configuration of software or hardware. Firefox browser is preferred because textures may not work properly in Google Chrome if its security policy prevents loading texture files locally. You should extensively test the animation controls to ensure that any control action will not cause the system to freeze, crash, or any scene objects to disappear or behave in a strange way.

### Deliverables

1. The source code of the entire WebGL application and any necessary supporting files such as libraries and textures. The program code should be suitably commented and come with necessary instruction for using it.
2. An electronic copy of a short report (no more than 1000 words) that documents the design/implementation decisions, difficulties, or problems (if any) and evidence and/or conclusions of test and evaluation of the application against the specification.

## Appendix 1

### Task 1: Camera system configuration



### Task 2: Illustration of the shape of the satellite (not for the correct scale or accurate colours)

Assignment Project Exam Help



## Appendix 2 Marking Scheme

Task One (50%)		
<b>Correct detection of traffic</b> <ul style="list-style-type: none"> <li>Speeding detection (30mph).</li> <li>Oversize vehicle detection (2.5m).</li> <li>Fire engine detection (colour&amp; size ratio).</li> </ul>	15	
<b>Feature selection and detection</b> <ul style="list-style-type: none"> <li>Selection of appropriate features.</li> <li>Correct detection methods/procedures &amp; appropriate parameters.</li> <li>Quality of extracted features.</li> </ul>	10	
<b>Testing &amp; Evaluation</b> <ul style="list-style-type: none"> <li>Test against all use cases and test script</li> <li>Evidence &amp; documentation (screenshots and discussions) of test against traffic scenarios</li> </ul>	10	
<b>Report</b> <ul style="list-style-type: none"> <li>Discussion of application conditions and assumptions</li> <li>Feature selection &amp; justification of processing methods</li> <li>System diagram</li> <li>Evaluation, test script &amp; evidence (output, screenshots)</li> </ul>	15	
Task Two (50%)		
<b>Object models</b> Accurate models of each element of the scene.	13	
<b>Surface attributes</b> Colours and/or texture mapping.	7	
<b>Lighting</b> Use of light and any justification	7	
<b>Animation and interactive parameter control</b> Overall animation, satellite orbit & speed control	12	
<b>Scene navigation control</b> Translation & rotation controls	6	
<b>Test &amp; evaluation</b> Report	5	