Setting up the software: To run this application, the system must have MS Visual Studio IDE installed and OpenCV integrated with it. It has been developed and tested using Visual Studio 2008 with OpenCV 2.4.4 but should with other versions of OpenCV too. However, there is some unresolved compatibility issue related to the BGS library that prevents it from compiling successfully under Visual Studio 2010. No testing has been done with earlier versions of Visual Studio but it is unlikely to run with those either.

The application has been tested with and runs fine under both Windows XP 32 bit and Windows 7 64 bit operating systems. No testing has been done under any version of Linux.

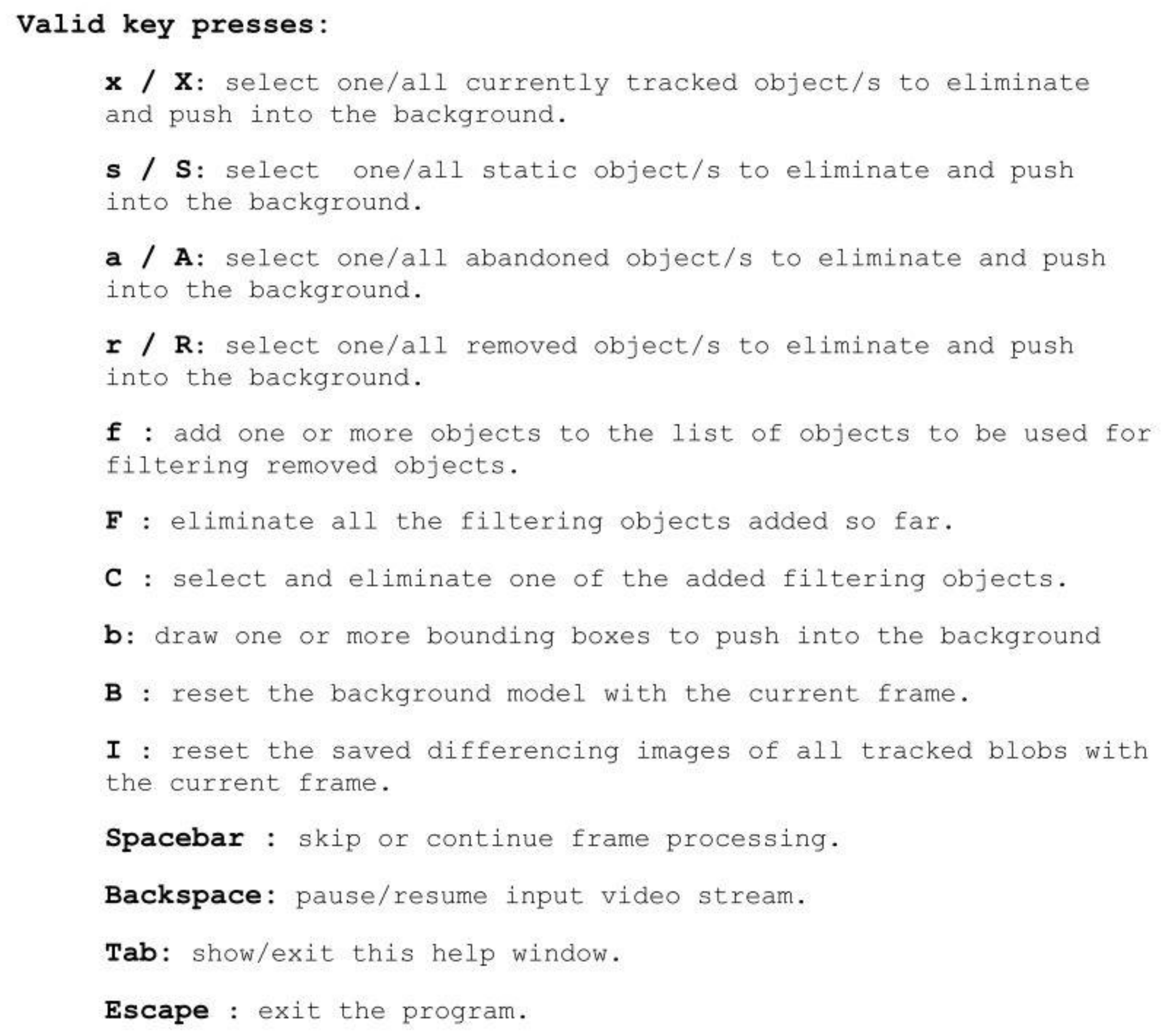
After installing Visual Studio 2008 and OpenCV 2.4.4 (both included in the Software directory), use the steps outlined at http://docs.opencv.org/doc/tutorials/introduction/windows_install/windows_install.html to complete the integration process. Use the **Installation by Using the Pre-built Libraries** option**.**

After this is completed successfully, open the file called abandoned_object_detection.sln present in the Code directory (using File→Open→Project/Solution). The actual source code and Visual C++ project files are located in the subfolder called `abandoned_object_detection`. The other subfolder called `blob_detection` holds the blob detection library called `cvBlobsLib` and should be left in the same relative position to the remaining code for it to compile. After the file is successfully loaded, use `Debug→Start Debugging (F5)` or `Debug→Start Without Debugging (Ctrl+F5)` to execute the application.

Setting up input datasets: This program takes its input either from a camera or from a video file. The latter are to be kept in the `abandoned_object_detection/input` directory under separate subfolders for different datasets. Currently there are four datasets with corresponding folders called `AVSS, PETS2006, PETS2007 and Custom`. The video files in `PETS2006` and `PETS2007` datasets follow a naming template: `set_x1_view_x2_size_x3.avi` where `x1` is the set ID (0-6 for `PETS2006` and 0-9 for `PETS2007`), `x2` is the camera view ID (0-3) and `x3` is the size ID (0-2 with 2 being the original resolution (720x576), 1 being half of it (360x288) and 0 being a quarter (180x144)). Note that the size 0 files are not included by default and need to be created using a suitable video encoder (`MediaCoder` is included here for this purpose). In case the program encounters problems in reading the input video file, ensure that the concerned codecs are installed in the system (the `K-Lite Mega Codec Pack` included here should solve any such problems).

Specifying initial parameter values: The initial parameter vales can be specified in a formatted text file called `init_parameters.txt` present in the `abandoned_object_detection` subfolder. Each line of this file can contain either a comment, blank space or a comment followed by a numeric parameter value. Comments are specified by surrounding them with #. No line should contain comments after a parameter value (or any non-comment string) has already been specified in that line. The parameters in the default version of this file are grouped by the module they pertain to and each is preceded by a comment describing it and specifying its valid values. Each parameter is also numbered and its number should match with the corresponding number in the function called `readInitial**Params` in the file called `systemParameters.cpp` where `**` is the name of the respective module.

Interacting with the program: Once the program has begun executing, any of its parameter values can be changed in real time by adjusting the relevant track bar position in the relevant window It should be noted that since OpenCV only allows integer values on track bars, all floating point values are multiplied by a suitable multiple of 10 for the purpose of putting on the corresponding track bar. In addition, the program accepts several keyboard inputs whose details can be obtained any time by pressing the Tab key. These are also presented here for convenience:



Valid key presses:

x / X: select one/all currently tracked object/s to eliminate and push into the background.

s / S: select one/all static object/s to eliminate and push into the background.

a / A: select one/all abandoned object/s to eliminate and push into the background.

r / R: select one/all removed object/s to eliminate and push into the background.

f : add one or more objects to the list of objects to be used for filtering removed objects.

F : eliminate all the filtering objects added so far.

C : select and eliminate one of the added filtering objects.

b: draw one or more bounding boxes to push into the background

B : reset the background model with the current frame.

I : reset the saved differencing images of all tracked blobs with the current frame.

Spacebar : skip or continue frame processing.

Backspace: pause/resume input video stream.

Tab: show/exit this help window.

Escape : exit the program.

Each window also has its own help window that can be invoked by a left click anywhere inside that window and removed by a right click. All of these help files are present under `abandoned_object_detection/help` directory as JPEG image files.

Following is a list of the names of windows created by the various modules:

- User Interface module: `Current Frame` and `Combined`
- Pre-processing module: `Pre-processing Parameters`
- BGS module: `BGS parameters`

- Foreground Analysis module: `Foreground Analysis Parameters` and `Foreground Processing Toggles`
- Blob extraction module: `Blobs`
- Blob tracking module: `Blob Tracking Parameters` and `Blob Matching Parameters`
- Abandonment analysis module: `Abandonment Analysis Parameters`
- Blob filtering module: `Blob Filtering Parameters`

Getting output: Apart from providing real time output in its many windows, the application also saves a few selected outputs as AVI video files under the `abandoned_object_detection/output` directory using the same naming scheme as the input file. The outputs of different datasets (and camera input) are saved under respective separate subfolders. By default the following three output videos are created for each input file (or camera):

- **<input file name>_ bounding_boxes.avi:** shows the bounding boxes of all the tracked blobs visible in the current frame.
- **<input file name>_ combined.avi:** shows four images: current frame, current background model, foreground mask produced by BGS module, refined mask produced by the foreground processing module and the result of abandonment analysis module (either region growing or edge detection) on both current and background images.
- **<input file name>_ static_mask.avi:** shows a mask of the currently tracked and visible objects that is fed back to the BGS module to enable object level background updating.

These outputs can be changed by simply updating the concerned `CvVideoWriter` object in the main function (`main.cpp`)