Image Processing in UAV, UGV for Search and Rescue Prototype.

Vishwas K.M
University of Plymouth
Drake Circus
Plymouth, PL4 8PW
vishwas.kalipalyamruthyunjaya@pos
tgrad.plymouth.ac.uk

ABSTRACT

This report explains the image-processing task that was integrated into search and rescue prototype project. The report covers the background of image processing and the tools used. Explains the tools used for developing the system, and evaluation of the system. At the end, discussion on how the system can be further developed is explained.

Keywords

Image processing, UAV, UGV, OpenCV, Cascade classifier.

1. INTRODUCTION

This is a digital age where information and technology are gold. Abundance of technology exists that makes our daily life easier. A natural disaster circumstance, an important area, where technology's help is invaluable is in. natural disasters are destructive and calamitous. However, preventing such disasters has a bleak chance, but its impact is reducible. What are the impacts? The direct impacts are environmental damage, people's life, climate, etc. As mentioned, not all the disasters are preventable, but there are ways to act quickly to help victims of such disasters. Hence, the importance for search and rescue is of the highest order. There are considerable approaches to execute. However, in this project prototype Marc Glover, Devdatta Narote, and I have planned an idea as to how we approach the natural disaster search and rescue operations by interacting with UAV and UGV. The project not only aims at natural disaster search and rescue alone, also extends its reach for situations such wildlife monitoring, security maintenance in controlled plant/highly secured area, and terrorism.

The very idea of using unmanned aerial vehicle (UAV) and unmanned ground vehicle (UGV) is to do the search task by UAV and rescue/help operation by UGV. How do we approach this? Answer is, by interaction. For an interaction between UAV and UGV there must be signal/information exchange between the two. Moreover, the search operation system, UAV, sends the command to rescue operation system, UGV, to execute an action. The UAV sends the command to UGV when the operator/drone identifies the casualty. So, how do they find casualty? Marking a casualty is easy even with the naked eye. However, in real-time application the operator needs to concentrate on drone control while marking the casualty. To help the operator, the system can include image-processing techniques to identify the casualties.

This prototype aims at tracking humans that would demonstrate the use of image processing and its help in this search and rescue concept of ours.

2. KEY PLAYERS

Some of the major companies that involve in smart image processing and its usage in multiple fields for various purpose are Machine Vision Technology Ltd, Gentex Corporation, Honeywell, and NVIDIA.

3. SYSTEM DEVELOPMENT

3.1 Programming Tools:

OpenCV is an open source library created in the late 1990's by Intel to perform advanced vision based processes such as motion and object detection and recognition. The cross-platform library sets its focus on real-time image processing and includes patent-free implementations of the latest computer vision algorithms. OpenCV supports many algorithms related to Computer Vision and Machine Learning. OpenCV-Python is the Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language [1, 2]. Python is a general-purpose programming language started by Guido van Rossum, known for its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing any readability. [2] Numpy is a highly optimized library for numerical operations. It provides a MATLAB like syntax. The OpenCV array structures are converted to-and-from Numpy arrays. This makes any operation easy to combine with OpenCV [2].

3.2 Background

The image processing software can be implemented on different system, not exclusive for AR-Drone alone. Though hardware -- video camera -- dependent for processing live video, it can be implemented in systems where stored images and/or video feedback is available. Versatility makes this image processing cost-effective and reduces dependency overhead.

The work based on open source coding has its pros and cons. Advantages like licensing this product costs less, easily managed while able to work continuously on improvement in real-time. Disadvantages are developer centric: if part of the code is not made available to users due to patent or licensing, and users are sometimes confused with versions.

The work is independent of hardware and the setup is software based. The software needs a hardware system to run on and needs cameras for capturing the image. However, the hardware is a utility tool rather than developing software on the hardware. This makes the software cost effective. This project can be evaluated using other work already done in the field as a tool for comparison.

3.3 System

The body detection program built in python uses Haar Cascade Classifier. Like most of the feature recognition building process, this program involves stages such as acquiring the

image(s), image processing, distinctive character localization, template creation, and template matching in that order.

Image processing to template matching: The program first converts the colored frame captured into grey scale. Then, the cascade classifier uses the function detetctMultiScale() to detect objects of different sizes in the input image. Then, for every bodies, arms, legs, and faces detected using cascade classifier, invokes a draw rectangle function. Continuing, the eye cascade classifier within the face-detected boundary invokes draw rectangle function to draw boundaries around mouth and eyes.

Every rectangles/boundaries within the frame captured using the cascade classifier is displayed using imshow() function.

These rectangles drawn around the detected bodies acts as good signaling action for the drone operator to give attention and send out a signal and/or a command to UGV giving the GPS location. Then, the UGV will use GPS location data and move towards the received location.

To implement learning and training into this system, one needs to collect data set based on the images or the image that needs to be processed with the features that needs to be detected. Viola and Jones [3] used adaptive boosting (AdaBoost) technique to make the classifier learn and train. AdaBoost learning algorithm is used to boost the classification performance of a simple learning algorithm.

Once a network has been structured for a particular application, that network is ready to be trained. To start this process the initial weights are chosen randomly. Then, the training, or learning, begins. There are two approaches to training - supervised and unsupervised. Supervised training involves a mechanism of providing the network with the desired output either by manually "grading" the network's performance or by providing the desired outputs with the inputs. Unsupervised training is that the network has to make sense of the inputs without outside help [4].

4. TESTING AND EVALUATION

The complete prototype testing was conducted on Plymouth university campus.

The image processing system was developed using python programming first and tested on video capture using a webcam. and then on the video file. It was observed that detects noise due to lack of training the classifiers, but also works properly. In addition, with more training using machine-learning technology the processing and detection gets smoother and faster.

The image processing system after the initial testing and working was integrated with the AR-Drone C++ code because; the drone controller system was based on C++. However, there were difficulties with integration due to '.dll' files and compatibility.

The image processing results were distinguishable and effective. The initial target of the system was achieved successfully. Some of the results of the image processing are shown in the figure (1 - 3) [5].

From figure 2 we can observe that the cascade classifier used in the system detects noise along with other body detection. The reason is that, the video streaming is faster and the quality is average. These factors affect the system to process frames. This can be overcome using graphic accelerators and cameras with higher resolution.

Figure 1: detecting upper body even when the whole body is not visible.

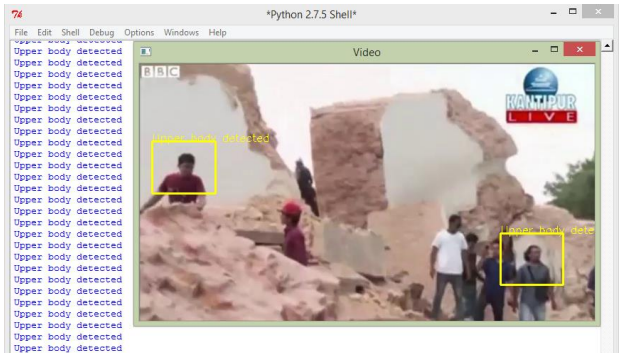

Figure 2: detecting multiple people and parts of body

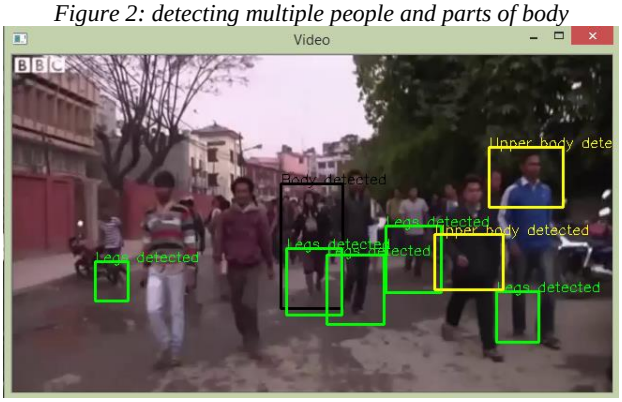

Figure 3: detection of noises along with actual body detection

5. FURTHER DEVELOPMENT

The system can be developed with enhanced image processing techniques such as identifying heat signatures, skin detection, and classifier for different body parts; in case of upper body cascade classifier, the classifier works only for upper body inclusive of arms, else the classifier cannot recognise the upper body. So, to detect different body parts effectively for cases

where the bodies are covered with debris, the specific body parts classifier helps the processing and the job for search and rescue operations easier and helpful.

6. REFERENCES

[1] OpenCv tutorials. Available at: https://opencv-pythontutroals. readthedocs.org/en/latest/py_tutorials/py_setup/py_intro/py_intro.html# intro. (Accessed: 24-Apr-15)

[2] OpenCV docs. Available at: http://docs.opencv.org/modules/contrib/doc/facerec/tutorial/facerec_video_recognition.html?highlight=haar http://docs.opencv.org/modules/contrib/doc/facerec/facerec_api.html#voidFaceRecognizer::train(InputArrayOfArrays src, InputArray labels) = 0 http://docs.opencv.org/master/d7/d8b/tutorial_py_face_detection.html http://docs.opencv.org/modules/contrib/doc/facerec/facerec_tutorial.html http://docs.opencv.org/doc/user_guide/ug_traincascade.html#viola2001 http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html#cascadeclassifier-detectmultiscale (Accessed: 24-Apr-15)

[3] Paul Viola, Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511-518.

[4] Psych, Available at: http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural3.html. (Accessed: 30-Apr-2015).

[5] AINT510, Available at: https://www.youtube.com/watch?v=HQN6X_G8Et8 (Accessed: 30-Apr-2015).