Autonomous Aerial Mapping and Observation Task for IMAV2014 Competition

Emmanuel Colles[1], Fabien Nollet[1], Bertrand Vandeportaele*[2,3], and Gautier Hattenberger†[1]

[1]ENAC, MAIAA, University of Toulouse, Toulouse, France
[2]CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[3]Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France

ABSTRACT

The IMAV2014 Competition gives an important place to computer vision and aerial mapping. This paper presents the different strategies and algorithms that have been developed in order to tackle these particular problems. The first task consists in automatically reading the digit displayed by a seven segment panel on the front wall of a building. Robust algorithms for segments detection are used in order to extract the features from the scene. The second task requires to produce an aerial map of the competition place. In order to generate a coherent composite image, photogrammetry methods are used, including interest points pairing and bundle adjustment. We propose to mix 3D mapping methods and standard 2D image mosaicking in order to achieve the processing in the time available during the competition while obtaining an geometrically consistent map.

1 INTRODUCTION

The IMAV competitions are always pushing the limits in terms of micro air vehicle performances and autonomy. The 2014 edition gives an important place to real-world operation by setting the mission in a real military training village. Since many tasks require the use of computer vision, the reliability and the robustness of the algorithms are a key point. This article presents the strategies and the algorithms developed to tackle two particular tasks: detecting and reading a seven segment digit placed on a building front wall; and making an aerial mapping of the competition area.

For the first task, presented in section 2, several methods are available to extract a digit and recognize it. The selected solutions are using the knowledge on the digit color and shape, with a robust segment detection[1] and the use of an optical character recognition. The extracted pattern is then compared to a database to find out the recognized digit.

The aerial mapping task, developed section 3, is based on photogrammetry methods[2] and use some of the MICMAC tools[3] and a new mapping framework that have been developped specifically for the occasion.

2 DIGIT RECOGNITION

Optical characters recognition is a problem that is studying in many fields as automotive to assist drivers by scanning road signs[4] , security to recognize license plates[5], verify signature on cheque or human assisting as well to read zip code and make easier letters sorting. For the IMAV2014 Competition, one of the mission elements consists in observing the longest sequence of digits on a panel placed on the front wall of a building. The digit are going to change with a rate of thirty seconds. So there will be 60 different digits during the thirty minutes of the IMAV Competition.

At this point, it is not sure if the UAV will be landed or hovering in stable flight to manage this mission. With this constraint in mind, different solutions are developed to insure the best digit recognition depending of the UAV position and stability landed or not. In this perspective, section 2.2 present a process providing efficient results if the UAV is landed and, in section 2.3, a second process that should be used if the UAV is flying but stable.

2.1 Preprocessing

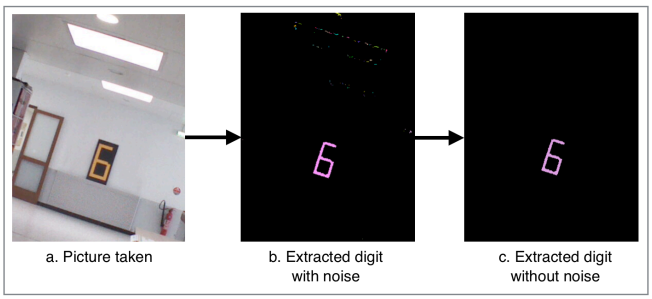

Figure 1: Pre-processing to extract digit bars

Although the end of the process is different, pre-processing is shared by both methods. This pre-processing exploits digit color and geometry of the pattern to read. By the use of the color, digit's bars are extracted from the image. In order to improve robustness, the HSL[1] color space

*bertrand.vandeportaele@laas.fr
†gautier.hattenberger@enac.fr

[1]Hue Saturation Lightness

is preferred to RGB[2] representation. This allows extraction of pixels of a certain hue, quite independently of lightning conditions, which is especially useful for outside missions. Further to this extraction, some noise can interfere and a process that exclude isolated pixels or small sets of pixels based on labelisation is used. As shown on figure 1, digit's bars are correctly extracted, and the next step, digit recognition, can be carried out according to the current UAV state.

2.2 *UAV landed*

In this section, the UAV is considered landed and its camera is observing the whole digit panel even if the digit is on a picture corner. Due to the condition that the UAV is not moving, the digit panel should project in the same position in the different images grabbed from the camera. In this case, the processing is done on an integrated picture. This process allows to sum pictures and highlight differences in a sequence. If each bar has been turned on one time at least as shown figure 2, the complete 8 digit can be rebuild. When the 8 digit is rebuilt, the four corners are extracted by computing intersections between lines detected robustly using the RANSAC[3] algorithm[6], independently of the panel orientation.

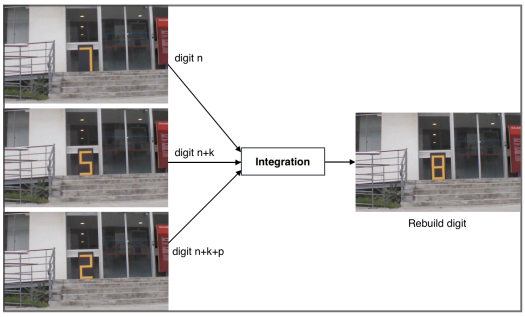

Figure 2: Integration of pictures to rebuild an 8 digit (all segments turned on)

Further to the extraction of the four corners, a bounding box is determined and a perspective transformation (homography) is applied to reshape the extract digit as shown figure 3 such that it fills an image of known size. This image is then compared to templates for recognition. Thanks to the homography, the process of decoding the digit is simplified, as it just consists in comparing the pixels values from the rectified image and an bank of ten reference images.

After this first extraction, it is not required to repeat the whole processing on the next picture, since the UAV is not moving. According to this strong constraint, the same corners are used to apply the homography to extract the next digit. Thus, processing will be shorter and with a reduced computational cost.

[2]Red Green Blue
[3]RANdom SAmple Consensus

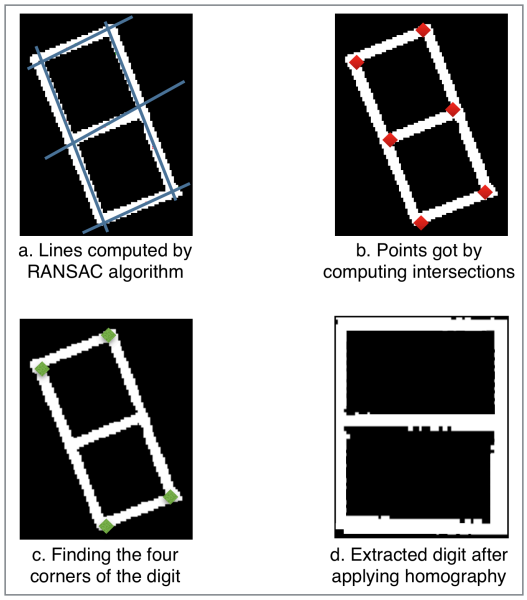

Figure 3: Extraction of the digits bounding box and application of an homography to get an image that does not depend on the viewing conditions

2.3 *UAV in stable hovering flight*

In this section, the case of the UAV flying in front of the digit panel is considered. Even if the UAV is very stable, it cannot be considered static. Processing presented in section 2.2 cannot work in this case because the embedded camera is a rolling shutter camera. This kind of camera acquires the image one line after the other and as the camera is moving during the process, the different lines are acquired with different camera positions and orientations. Under these conditions, lines for the scenes are not projected in straight lines in the images, and lines generally appear curved due to the rolling shutter distortions. An algorithm of line feature extraction in the image like the RANSAC based one used in the previous section will probably fail. For these reasons, another approach should be used.

The pre-processing detailed section 2.1 is kept as a first step. After this, an edge detector is applied to extract the skeleton of the extracted shape as shown figure 4. This skeleton allows to take out special features as extremities from the shape, intersections of segments and also if there are loops into the shape. With these three features, shown figure 5, it can be determined which digit is represented by the current extracted shape. If these features are not sufficient, extra specific processings can be used to find the correct digit from the shape as shown with the decision tree on figure 6.

There is no guarantee that the decision tree will give the correct answer. But since the computation time is fast enough (a few seconds on a standard laptop), several frames can be processed for the same digit (it changes every thirty seconds).

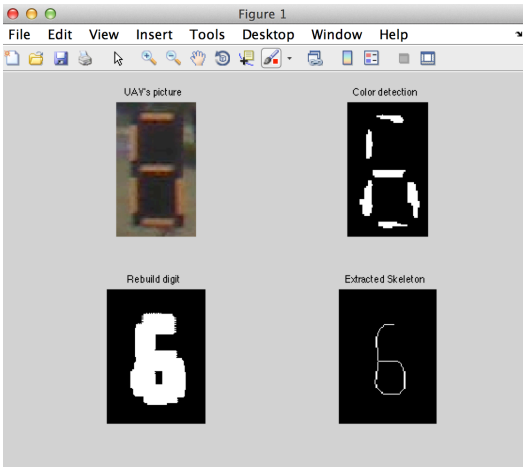

Figure 4: Extraction of the shape' skeleton

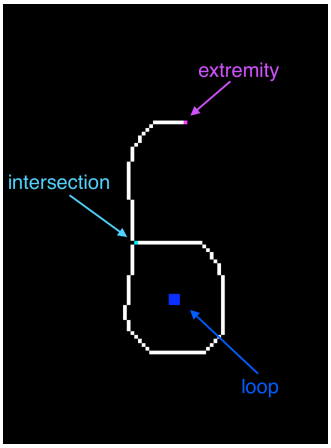

Figure 5: Specific features find out thanks to the shape' skeleton

The final decision can be made after a vote from the all the results.

3 Aerial Mapping

The aerial mapping mission consists in producing a mosaic of aerial photographies and then building the map of the village using photogrammetry[2].

3.1 Hardware

To take the aerial photographies, a fixed-wing UAV carries an embedded camera HackHD[7] synchronized with a GPS receiver module and the inertial measurement unit of the UAV. The HackHD camera is a miniature device (see figure 7) with a wide angle lens and that can take pictures with a resolution up to 9 Mega Pixels.

The GPS and IMU are used to determine the position and orientation of the camera while it is grabbing pictures of the

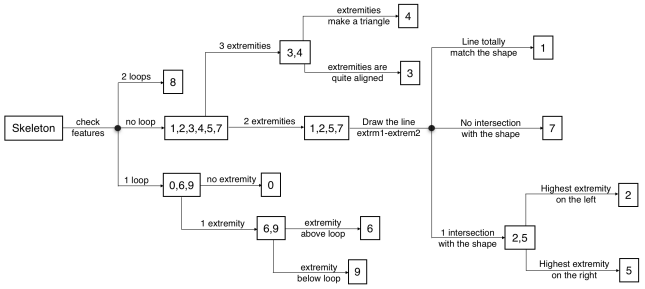

Figure 6: Path followed by the process according to special features extracted to recognize digits

scene. The tight synchronization allows to reduce the computation time of the mapping process by providing near real initial position and orientation of the camera to the photogrammetry algorithm. Also, it allows to create a georeferenced map, id. a map where the correspondence between a position in the map and in the real world is known.

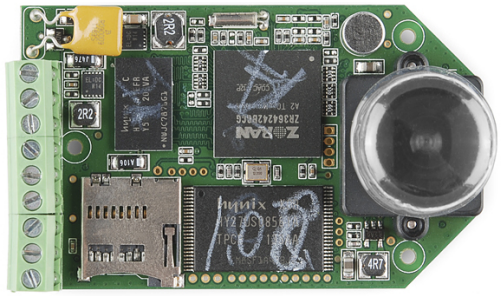

Figure 7: a camera HackHD

3.2 Orthophotomap creation

To create the map, the MicMac[3] program is used as a first step. It is an open-source software developed by the IGN[4] providing tools to compute tie points[5], digital elevation model and dense 3D points cloud from a set of overlapping images.

The workflow to compute an orthophotomap with this software is split in several steps:

Tie points computation : first, feature points in the aerial photographies are extracted and then matched between pairs of images (tie points[8] computation),

Bundle adjustment[9] : the 3D position of each feature point and the position and orientation of the camera for each picture is computed,

[4]French National Institute of Geographic and Forest Information

[5]Tie points are feature points from several images that correspond to the same point of the scene

Dense 3D points cloud : afterward, a dense 3D points cloud of the scene using stereoscopy principles is done,

Orthophotomap : finally, the 3D points cloud is projected onto the horizontal plane, and thus the desired orthophotomap is obtained.

However, this approach requires long computation time (about dozens of hours), mainly due to the computation of the dense 3D points cloud. We propose a strategy that allows to shorten the required computation time. From the original photogrammetric process, we replace the third stage by a less precise one which assumes that the environment is locally planar. This allows to avoid the explicit 3D computation of the altitude for the ground at every pixels of the orthophotomap by linearly interpolating it from a simple 3D model named a Digital Terrain Model. The DTM is a model made of a 2D cartesian grid with an associated altitude for each cell.

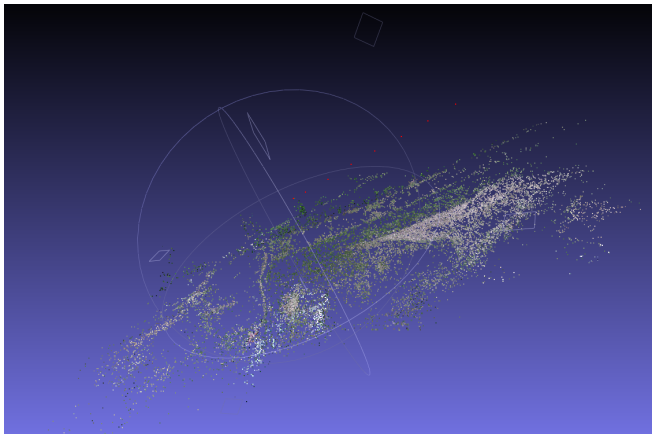

Figure 8: The sparse 3D point cloud obtained from the MICMAC tools.

The two first stages of the MicMac software are used to get the sparse 3D point cloud of the feature points. Such a 3D point cloud is shown in the figure 8. A DTM grid is then adjusted to this point cloud by robust least squares fitting. The resulting model, as shown in the figure 9, exhibits a regular mesh structure made of triangles. The robust estimation allows to compute the DTM without corrupting it because of 3D points that were too noisy or even completely wrongly estimated. It also allows to reject the 3D points that were lying on objects above the ground. Finally, the use of the DTM model allows to fill the gaps of the model; if some elements in the grid do not correspond to any 3D feature points, then we interpolate the altitude of the grid element from its neighbors.

Once the grid has been estimated, the generation of the orthophotomap is quite simple. We achieve the same processing for each image grabbed by the camera and then integrate the results. This allows to parallelize by processing different images on different processors and hence reduce the processing time.

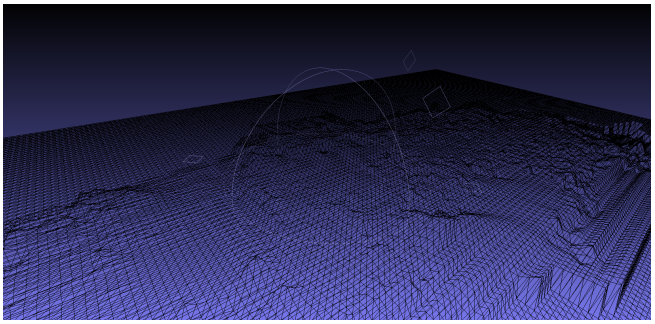

Figure 9: The fitted 3D grid used to model the terrain.

To process one image, each pixel of the associated orthophotomap is projected to the camera, given the position and orientation of the camera and the altitude corresponding to the pixel of the orthophotomap. If this projection lies inside the images boundary, then the orthophotomap is colored from the camera's image pixel. The alpha channel in the image is used to indicate if the pixel of the orthophotomap has been colored or not. The figure 10 shows on image acquired by the camera and the figure 11 shows the corresponding orthophotomap.

Once a sufficient number of images have been processed, a global orthophotomap is generated as shown in the figure 12. As all the orthophotomaps are georeferenced, it is possible to add new data to the global orthophotomap afterwards, for instance to keep the map up to date.

Figure 10: An image from the camera used to construct the map.

Figure 11: The image from figure 10 projected on the 3D map.

Figure 12: The result of the projection of ten images on the 3D map.

4 Conclusion

In this paper, the methods for solving two of the IMAV2014 mission elements are presented. The first one concerns an automatic digit detection and recognition. Since all the aspects of the mission are not known, several algorithms are investigated in order to get the most robust solution on the day of the competition. The second mission element consists in making a map of the flight area, by stitching several pictures. An off-line solution has been developed using images and the associated 3D poses acquired by GPS and IMU. The proposed solution is using bundle adjustment in order to improve the alignment of the images. As processing time is a key concern during the competition, a compromise has been made. The proposed solution provides better results than simple 2D images mosaicking and it doesn't require as much time as the complete 3D orthophotomap generation thanks to the interpolation of altitudes in the DTM model.

References

[1] Andrea Fusiello Roberto Toldo. Robust multiple structures estimation with j-linkage. http://www.diegm.uniud.it/fusiello/papers/eccv08.pdf, October 2008.

[2] T. Schenk. Introduction to Photogrammetry. http://www.mat.uc.pt/~gil/downloads/IntroPhoto.pdf, 2005.

[3] IGN. MicMac. http://logiciels.ign.fr/?-Micmac.

[4] Mutsumi Omori Tetsushi Koide Masaharu Yamamoto, Anh-Tuan Hoang. Speed traffic-sign recognition algorithm for real-time driving assistant system speed traffic-sign recognition algorithm for real-time driving assistant system speed traffic-sign recognition algorithm for real-time driving assistant system. http://sasimi.jp/new/sasimi2013/files/pdf/p195_R3-11.pdf.

[5] A.Srisaila S.Kranthi, K.Pranathi. Automatic number plate recognition. http://digitalpdf.org/automatic-number-plate-recognition-download-w79 July 2011.

[6] Kenney-C.S. Manjunath B.S. Zuliani, M. The multi-ransac algorithm and its application to detect planar

homographies. http://vision.ece.ucsb.edu/publications/05ICIPMarco.pdf, September 2005.

[7] HackHD camera. http://www.hackhd.com/.

[8] Kourosh Khoshelham. Role of Tie Points in Integrated Sensor Orientation for Photogrammetric Map Compilation. http://www.asprs.org/a/publications/pers/2009journal/march/2009_mar_305-311.pdf, march 2009.

[9] Richard Hartley Bill Triggs, Philip McLauchlan and Andrew Fitzgibbon. Bundle Adjustment — A Modern Synthesis. http://lear.inrialpes.fr/pubs/2000/TMHF00/Triggs-va99.pdf, 2000.