Can Speedup Assist Accuracy? An on-board GPU-Accelerated Image Georeference Method for UAVs

L. Bampis, E. G. Karakasis, A. Amanatiadis, and A. Gasteratos

Department of Production and Management Engineering,
Democritus University of Thrace, Xanthi, Greece

Abstract. This paper presents a georeferenced map extraction method, for Medium-Altitude Long-Endurance UAVs. The adopted technique of projecting world points to an image plane is a perfect candidate for a GPU implementation. The achieved high frame rate leads to a plethora of measurements even in the case of a low-power mobile processing unit. These measurements can later be combined in order to refine the output and create a more accurate result.

1 Introduction

Remote sensing Unmanned Aerial Vehicles (UAVs) are increasingly used in various civilian domains such as agriculture, environmental monitoring, wildfire detection and urban geodata tasks. High-accuracy georeferenced mosaics are required in many remote sensing applications such as coastal erosion monitoring, post-disaster assessment, micro-topography and urban geoinformation [10]. The precision and correctness of the models however, are affected by many parameters such as the accuracy of the on-board inertial measurement units, the Global Positioning Systems (GPS) and the optical image sensors.

Image georeferencing has extensively been investigated in the last decades. A variety of approaches have been introduced in the literature which address this issue using the flat world assumption [8, 13]. Such kind of methods are limited to high flight altitudes or planar terrains. In order to create a georeferenced map for terrains with rough geomorphology one could distinguish two main trends: 1) algorithms that derive information about the world morphology from Digital Terrain Models (DTMs) or Digital Surface Models (DSMs) [3, 1, 11] and 2) techniques that are based on the correlation between the optical information from consecutively acquired images [12, 6, 10]. In addition to the aforementioned trends, there is also an increased interest in accelerating the production of georeferenced maps using GPGPU computing [11].

Taking advantage of the continuous progress in the precision of the aforementioned sensors, recent sophisticated computer vision algorithms can provide in real-time, significantly improved accuracies in high resolution spatial data. This improvement however, comes with a critical trade off, which is the necessity for more computational power on the on-board processing units, especially

when on-board real-time processing is required. Since power consumption and weight limitations are two of the most fundamental constraints in a UAV design, an approach of utilizing low-weight and low-power processing boards is highly essential.

In this paper, a method for real-time on-board georeferenced map extraction for Medium-Altitude Long-Endurance (MALE) UAVs is proposed, based on the projection of the world points to the grabbed camera frames. The algorithm extends well on a GPU, providing the possibility of high processing frequencies. The achieved frame rates offer multiple measurements which can be used to increase the accuracy of the final extracted map.

2 Proposed Method

In this section the proposed method, which is based on GPS and Inertial Measurement Unit (IMU) sensors as well as on terrain information, using available Digital Elevation Models (DEMs), is presented. Through a process of a forward and backward projection between UAV camera sensor images and the used DEMs, the method produces a colored and georeferenced 3D point cloud. An orthographic projection of this point cloud to the image plane produces an orthorectified, georeferenced image. At this point it should be noted that the z-axis of the camera is always pointing to nadir, so as its image plane is parallel to the ground.

According to Karakasis et al. [5], a DEM produced by a fusion process between different DEMs, like SRTM and ASTER, is characterized by more accurate elevations. Since the proposed method depends on the sensors and DEMs accuracy, a fused version of ASTER and SRTM is selected to be used.

Although basic in principle and sensitive to GPS, IMU and DEMs accuracy, the proposed algorithm is a perfect candidate for a GPU implementation, since its mathematical foundation is consistent with that of graphics and thus, a great acceleration can be achieved. The advantage of speeding up the proposed algorithm is that it increases the measurements density for the 3D point cloud by allowing a corresponding increase of the camera's frame rate. This fact lead us to conclude that we can reduce the overall error of georeference by averaging measurements which correspond to the same world point. Furthermore, it is worth noticing, that the suitability of the proposed methodology for a GPU implementation, allows the use of cheaper and less power consuming equipment.

2.1 DEMs Fusion

A four-step fusion process based on the Chebyshev spectral expansion [5], is used in order to produce a fused version of the freely available ASTER and SRTM DEMs. The fusion steps are the following: 1) the preprocessing process, which addresses resolution issues of the used DEMs, 2) the forward transform of the DEMs using the Chebyshev spectral expansion, which produces a set of spectral expansion coefficients, 3) the weighted average of the expansion coefficients,

which produces a set of fused coefficients and finally, 4) the inverse spectral transform of the fused coefficients in order to produce the fused DEM.

2.2 Mathematical Foundation and Error Reduction

The provided measurements from the GPS and IMU can be represented using two vectors, namely a 3-dimensional for the position and a 4-dimensional quaternion for the orientation:

$$ {}^{uav\_m}_{w}\boldsymbol{p}^{(i)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}^{uav\_m}_{uav\_t}\boldsymbol{p}^{(i)} + {}^{uav\_t}_{w}\boldsymbol{p}^{(i)} \qquad and \tag{1} $$

$$ {}^{uav\_m}_{w}\boldsymbol{q}^{(i)} = \begin{bmatrix} k_x \sin(\theta/2) \\ k_y \sin(\theta/2) \\ k_z \sin(\theta/2) \\ \cos(\theta/2) \end{bmatrix} = {}^{uav\_m}_{uav\_t}\boldsymbol{q}^{(i)} \otimes {}^{uav\_t}_{w}\boldsymbol{q}^{(i)} \tag{2} $$

where $\hat{\boldsymbol{k}} = [k_x \; k_y \; k_z]^T$ is a unit vector on the rotation axis, while θ is the rotation angle. The vector ${}^{uav\_m}_{w}\boldsymbol{p}^{(i)}$ represents the measured and ${}^{uav\_t}_{w}\boldsymbol{p}^{(i)}$ the true position of the UAV with respect to the global frame of reference, while ${}^{uav\_m}_{uav\_t}\boldsymbol{p}^{(i)}$ is an error factor introduced by the GPS. Similarly, ${}^{uav\_m}_{w}\boldsymbol{q}^{(i)}$ represents the measured and ${}^{uav\_t}_{w}\boldsymbol{q}^{(i)}$ the true orientation of the UAV with respect to the global frame of reference, while ${}^{uav\_m}_{uav\_t}\boldsymbol{q}^{(i)}$ is a quaternion representing the error introduced by the IMU. It is safe to assume that both error parameters follow a zero mean Gaussian distribution [9]: ${}^{uav\_m}_{uav\_t}\boldsymbol{p}^{(i)} \sim N(0, Q_T)$ is the error of translation with covariance matrix Q_T and $\theta_e^{(i)} \sim N(0, \sigma_R^2)$ the error angle corresponding to ${}^{uav\_m}_{uav\_t}\boldsymbol{q}^{(i)}$ with σ_R^2 variance. Additionally, we introduce the expression $\boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}^{(i)}\right)$ as the rotation matrix corresponding to the quaternion ${}^{uav\_m}_{w}\boldsymbol{q}^{(i)}$. Note that the superscript (i) is a notation indicating the measurements in different points in time.

On the other hand, every point in the world can be represented with respect to the global frame of reference as:

$$ {}^{w}\boldsymbol{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3} $$

and given the position and the orientation of the UAV, one can express a world point with respect to the UAV frame of reference as:

$$ {}^{uav}\boldsymbol{p}^{(i)} = \boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}^{(i)}\right) {}^{w}\boldsymbol{p} + {}^{uav\_m}_{w}\boldsymbol{p}^{(i)} \tag{4} $$

Finally, given the transformation between the GPS, IMU and camera (${}^{cam}_{uav}\boldsymbol{T}$), as well as the camera intrinsics matrix (${}^{img}_{cam}\boldsymbol{C}$), the point ${}^{uav}\boldsymbol{p}^{(i)}$ can be expressed

as coordinates on the camera frame using homogenous coordinates:

$$\boldsymbol{px}_L^{(i)} = \begin{bmatrix} u_L \\ v_L \\ w_L \end{bmatrix} = {}^{img}_{cam}\boldsymbol{C}^{cam}_{uav}\boldsymbol{T} \begin{bmatrix} {}^{uav}\boldsymbol{p}^{(i)} \\ 1 \end{bmatrix} = {}^{img}_{uav}\boldsymbol{C} \begin{bmatrix} {}^{uav}\boldsymbol{p}^{(i)} \\ 1 \end{bmatrix} \tag{5}$$

Using Eqs. (4) and (5), a color measurement for the world points inside (i)th's camera frustum can be obtained and assigned to a global image representing the georeferenced map as $I_G\left(\boldsymbol{px}_G^{(i)}\right) = I_L^{(i)}\left(\boldsymbol{px}_L^{(i)}\right)$. Furthermore, it is easy to extract the correlation between $\boldsymbol{px}_L^{(i)}$ and $\boldsymbol{px}_G^{(i)}$ on the common plane from an equation like:

$$\boldsymbol{px}_G^{(i)} = \begin{bmatrix} u_G \\ v_G \\ w_G \end{bmatrix} = \boldsymbol{Q}^{(i)}\ \boldsymbol{px}_L^{(i)} + \boldsymbol{J}^{(i)} \tag{6}$$

where the matrix $\boldsymbol{Q}^{(i)}$ and vector $\boldsymbol{J}^{(i)}$ are derived from the Eqs. (4) and (5) for a plane and contain information from ${}^{uav\_m}_{w}\boldsymbol{p}^{(i)}$ and ${}^{uav\_m}_{w}\boldsymbol{q}^{(i)}$ [13].

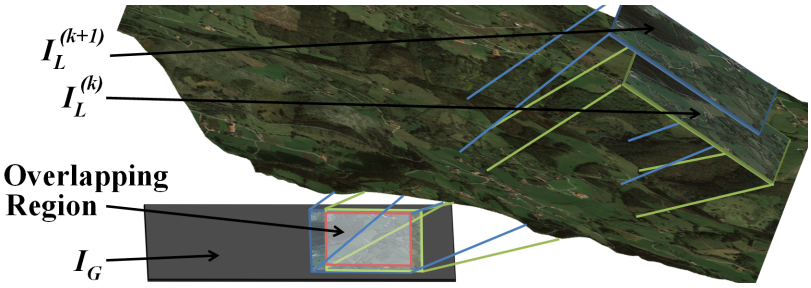


Fig. 1. Grabbed frames projected to a global georeferenced map through Eq. (6).

Fig. 1 illustrates the aforementioned procedure for an example of two grabbed frames. Images $I_L^{(k)}$ and $I_L^{(k+1)}$ are added to the global map I_G through the projection process encompassing though an error introduced by the IMU and GPS measurements.

Considering the noise reduction, instead of assigning one absolute color measurement on a world point, the plethora of available measurements is exploited. Thus, a better estimation for the georeferenced color of each world point can be obtained by taking the average of all the associated measurements.

The intuition behind that logic was the fact that noises introduced to the system by the measurement units have zero mean. With that in mind and from Eq. (1) we can easily prove that:

$${}^{uav\_m}_{w}\bar{\boldsymbol{p}} = {}^{uav\_m}_{uav\_t}\bar{\boldsymbol{p}} + {}^{uav\_t}_{w}\bar{\boldsymbol{p}} = {}^{uav\_t}_{w}\bar{\boldsymbol{p}} \tag{7}$$

where the notation $(\bar{})$ above the factors is used to describe their average value for all the acquired measurements over time.

For the rotation on the other hand, the small angle approximation needs to be considered. In general, if the rotation angle that a quaternion represents is small enough (like the case of noise); the quaternion can take the form:

$$\delta \boldsymbol{q} = \begin{bmatrix} \hat{\boldsymbol{k}} \sin\left(\delta\theta/2\right) \\ \cos\left(\delta\theta/2\right) \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \tag{8}$$

In that case the rotation matrix that corresponds to the quaternion is:

$$\boldsymbol{R}\left(\delta \boldsymbol{q}\right) \approx \boldsymbol{I}_{3x3} - \lfloor \delta\boldsymbol{\theta} \rfloor \tag{9}$$

where $\lfloor \delta\boldsymbol{\theta} \rfloor$ is the skew-symmetric matrix of the $\delta\boldsymbol{\theta}$ vector.

Taking into account Eq. (9), the rotation matrix $\boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}^{(i)}\right)$ given by

$$\boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}^{(i)}\right) = \boldsymbol{R}\left({}^{uav\_m}_{uav\_t}\boldsymbol{q}^{(i)}\right)\boldsymbol{R}\left({}^{uav\_t}_{w}\boldsymbol{q}^{(i)}\right) \tag{10}$$

and the fact that $\theta_e^{(i)}$ noise have zero mean, one can get:

$$\bar{\boldsymbol{R}}\left({}^{uav\_m}_{w}\boldsymbol{q}\right) = \overline{\boldsymbol{R}\left({}^{uav\_m}_{uav\_t}\boldsymbol{q}\right)\boldsymbol{R}\left({}^{uav\_t}_{w}\boldsymbol{q}\right)} = \bar{\boldsymbol{R}}\left({}^{uav\_m}_{uav\_t}\boldsymbol{q}\right)\bar{\boldsymbol{R}}\left({}^{uav\_t}_{w}\boldsymbol{q}\right) = \bar{\boldsymbol{R}}\left({}^{uav\_t}_{w}\boldsymbol{q}\right) \tag{11}$$

Starting from $\bar{I}_G\left(\boldsymbol{px}_G\right) = \bar{I}_L\left(\boldsymbol{px}_L\right)$, we need to associate the color information of the global map to its pixel coordinates. At this point, an assumption regarding the local luminosity distribution of an image needs to be introduced. We are going to assume that in a small global frame region, where the measurements for the same world point will fall, the image function will behave accordingly to the bilinear model:

$$I_G\left(\boldsymbol{px}_G^{(i)}\right) = diag\left(\begin{bmatrix} a \\ b \\ c \end{bmatrix}\right)\boldsymbol{px}_G^{(i)} \quad or \quad \bar{I}_G\left(\boldsymbol{px}_G\right) = diag\left(\begin{bmatrix} a \\ b \\ c \end{bmatrix}\right)\bar{\boldsymbol{px}}_G \tag{12}$$

or in other words that the color remains stable or relates bilinearly, with coefficients a, b and c, to the pixel coordinates, for a small patch of the image. The combination of the above with Eq. (6), relates the average of the global pixel coordinates with the local ones, while the $\bar{\boldsymbol{Q}}$ and $\bar{\boldsymbol{J}}$ factors exclude their noise as it was proven with Eqs. (7) and (11).

To test the aforementioned assumption's strength, an experiment was conducted where the image presented in Fig. 2(a) was trimmed into many subimages. A 20×20 neighborhood was extracted for every pixel of the image, as well as the neighborhood's best fitted plane accordingly to the luminosities. The extracted plane together with the corresponding luminosity values is shown in Fig. 2(b). Fig. 2(c) presents a histogram of the absolute differences between those planes and the corresponding luminosities, while Fig. 2(d) indicates the mean absolute difference for the individual coordinates of all tested patches. Since the error between the true pixel intensities and the assumed plane is sufficiently small, it is justified to carry on having in mind that Eq. (12) locally holds but it is expected to create a blurring effect to the produced map. Another approach

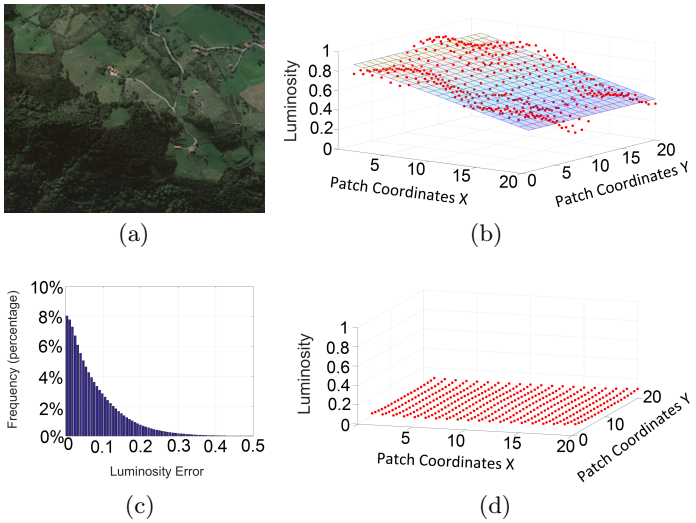


Fig. 2. (a) Tested image. (b) Extracted plane with corresponding luminosity values for a 20×20 size patch. (c) Histogram of the absolute differences between the plane and the luminosities for all the pixel of the tested image. (d) Average absolute difference for the 20×20 patch coordinates. Note that the range of the pixel intensities is [0 1].

which doesn't require this assumption, would be to keep track of the average of the local pixel coordinates and later combine it with the local pixel luminosities. That methodology, though better in principle, is computationally more expensive for a GPU implementation and that is why it was not initially adopted. However, an extension of the methodology including the aforementioned approach is in authors' future plans.

Continuing the derivations, from Eq. (5):

$$\bar{p\boldsymbol{x}}_{\boldsymbol{L}} = \overline{{}^{img}_{uav}\boldsymbol{C} \begin{bmatrix} {}^{uav}\boldsymbol{p} \\ 1 \end{bmatrix}} \quad or \quad \bar{p\boldsymbol{x}}_{\boldsymbol{L}} = {}^{img}_{uav}\boldsymbol{C} \begin{bmatrix} {}^{uav}\bar{\boldsymbol{p}} \\ 1 \end{bmatrix} \tag{13}$$

since the ${}^{img}_{uav}\boldsymbol{C}$ matrix is constant. Additionally, from Eq. (4) we obtain:

$${}^{uav}\bar{\boldsymbol{p}} = \overline{\boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}\right){}^{w}\boldsymbol{p}} + {}^{uav\_m}_{w}\bar{\boldsymbol{p}} \tag{14}$$

Once again, given that the position of the world points is not changing over time, the term ${}^{w}\bar{\boldsymbol{p}}$ can be substituted with ${}^{w}\boldsymbol{p}$. Furthermore, from the above equation, together with (7) and (11):

$${}^{uav}\bar{\boldsymbol{p}} = \overline{\boldsymbol{R}\left({}^{uav\_m}_{w}\boldsymbol{q}\right){}^{w}\boldsymbol{p}} + {}^{uav\_m}_{w}\bar{\boldsymbol{p}} = \bar{\boldsymbol{R}}\left({}^{uav\_t}_{w}\boldsymbol{q}\right){}^{w}\boldsymbol{p} + {}^{uav\_t}_{w}\bar{\boldsymbol{p}} \tag{15}$$

Finally, the combination of Eqs. (12), (13) and (15) provides a more accurate measurement for the georeferenced color of a world point, since the noise factors are eliminated.

At this point, the importance of a measurement plethora becomes clear. As the number of available measurements for a world point grows bigger, the average GPS and IMU input is better approximated and the noise is closer to zero. This result is verified in Subsection 4.3, where the accuracy improvement with respect to the growth of measurements for a world point is presented.

3 GPU Implementation

Inspired by modern graphics rendering techniques, we introduce an early step in the aforementioned algorithm. Instead of projecting all the individual world points of the area to be scanned on every grabbed frame from the UAV, we narrow down the world candidates by using an opposite projection. The technique of narrowing down the world candidates is well known in computer graphics, where the lighting is only calculated for the world points that can be seen by the camera. Therefore, in our case the pixels of each individual image are projected on a flat world [13], providing an estimation about the world points possibly inside the field of view of the camera. Only those points are projected again on the image plane and the ones that fall out of the image dimensions are rejected.

As it is stated before, the whole process of projection is a perfect candidate and can be efficiently implemented on a GPU. The process of assigning one new color measurement to every world point inside the camera's field of view, is passed on to GPU threads and executed in parallel. The transformation matrices are stored in the GPU constant memory, allowing all threads to access their values simultaneously. Furthermore, the DEM data are stored in the texture memory of the GPU, enabling fast hardware bilinear interpolation in order to achieve the desired analysis.

The GPU algorithm was implemented using nVidia's CUDA API [4] following its Single Instruction on Multiple Threads (SIMT) possibilities and constraints, while the reader is able to examine the specifics of the implementation online[1].

4 Experimental Results

This section aims to evaluate the proposed methodology in terms of both, computation time and accuracy of the resulted orthorectified and georeferenced map. As has already been mentioned, the GPU implementation of the proposed algorithm significantly accelerates the computation time leading to the conclusion that a faster camera could be used. High frame rates increase the density of the produced point cloud allowing a corresponding improvement of the overall resulting accuracy. Since, the construction of the UAV has not been finished yet, a simulation environment has been built in order to evaluate the algorithm's performance.

4.1 Simulation

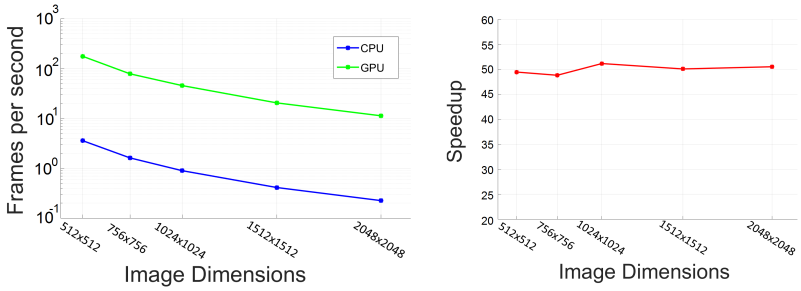
The simulated environment is based on the usage of DEMs and satellite images in order to construct a realistic scene. A dense lidar DEM dataset [7], which represents the area (*Longitude range:* $2^o24'00.0"W$ *to* $2^o25'48.0"W$ *and Latitude range:* $43^o18'00.0"N$ *to* $43^o19'48.0"N$), with resolution 1 meter is used as ground truth in order to evaluate the resulted orthorectified map of the proposed

[1] `http://tinyurl.com/CUDA-AerialImgGeoref`

algorithm. Furthermore, Gaussian noise is added to the position and orientation of the simulated UAV in order to further simulate the errors, which are owed to sensors (GPS & IMU) accuracy. This procedure creates a position error about ± 5 meters from the exact position, while such orientation error, so as to have at most 2^o degrees error from the vertical axis (nadir).

4.2 CPU vs GPU - Time Performance Comparison

In order to present timing evaluations and comparisons, a CPU version of the aforementioned methodology was also implemented. Both algorithms (CPU & GPU) where executed and timed on a nVidia Jetson TK1 board. This board is equipped with a Tegra K1 GPU and a quad-core ARM Cortex-A15 CPU. The low power consumption ($\sim 14W$ at most) and the lightness ($\sim 120gr$) of this board (crucial requirements for modern systems [2]) enables the processing to be done online on the UAV without the need of extra communications, hardware and flight load.



(a) (b)

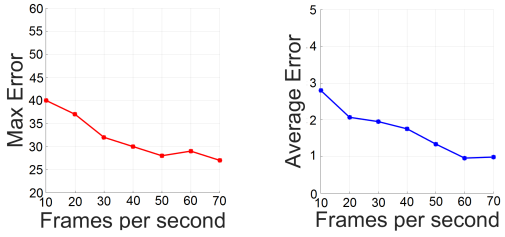
Fig. 3. (a) Fps comparison for the CPU and the GPU implementations (note the logarithmic scale). (b) GPU over CPU achieved speedup.

Fig. 3(a) presents the frames per second achieved for both CPU and GPU implementations over 5 different image resolutions, viz 512×512, 756×756, 1024×1024, 1512×1512 and 2048×2048. Real time performance was achieved for image up to 1512×1512 size. In general, modern cameras provide the possibility of choosing between several combinations of image resolution and frame rate. The most efficient choice for a real setup scenario would be the one that lies closer to a point of the presented graph. Fig. 3(b) on the other hand, depicts the speedup of the GPU over the CPU implementation. Note that the speedup remains constant as the resolution grows, proving a consistent scalability of the implemented algorithm.

4.3 GPU Implementation - Accuracy Evaluation

The number of measurements acquired for a world point depends on the factors of flight height and speed, the frame rate of the camera and the algorithm, as

well as the image dimensions. We choose to carry out the presented experiments with a setup where the UAV would fly at $1km$ height with $150km/h$ using a camera with 756×756 resolution, although our method is scalable for any other combination.



(a) (b)

Fig. 4. (a) Maximum measured error. (b) Average measured error. Note that the error scale is [0 255] for visualization purposes.

In order to relate the georeferenced map accuracy with the achieved speed of the algorithm, a variety of different frame rates were tested. The absolute difference between the ground truth georeferenced map and the map produced from the proposed algorithm (with introduced noise), is selected as a measure of error. As one can see in Fig. 4, the error is inversely proportional to the frame rate and consistently to the number of measurements for the individual points.

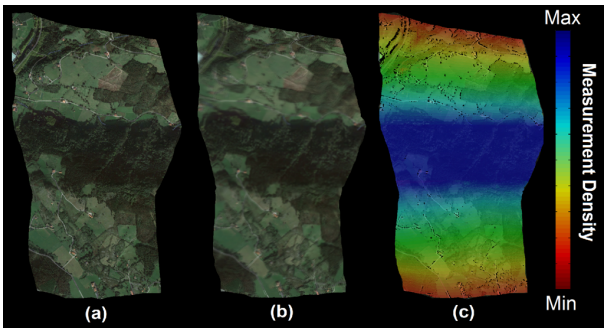


Fig. 5. (a) Georeferenced ground truth map. (b) Georeferenced produced map. (c) Measurements density of the scanned area.

Finally, the ground truth and the generated georeferenced-orthorectified map of the scanned area, can qualitatively be compared to each other in Fig. 5(a) and Fig. 5(b) respectively. Since the number of measurements for the points of the world is not uniformly distributed, ie the density is smaller at the beginning and the end of the UAV trajectory, only the region in the middle exhibits the optimum noise reduction. The measurement density is illustrated in Fig. 5(c), where the red coloring represents the territory with less measured world points, while moving towards blue the number of measurements grows bigger. At this point it should be noticed that although the output has blur appearance, the

georeference quality in the middle region is increased, since there are no ghost effects (as in the bottom and upper area) and the color information is better positioned in the world.

5 Conclusions

In this work, a GPU accelerated method able to generate a georeferenced map for a scanned area from a UAV, is proposed. Although the method is influenced by GPS and IMU noise, a high frame rate can be achieved by the algorithm (even for a mobile GPU), enabling the acquisition of a measurement plethora which subsequently was proven to create more precise results.

References

1. Amanatiadis, A., Karakasis, E., Bampis, L., Giitsidis, T., Panagiotou, P., Sirakoulis, G.C., Gasteratos, A., Tsalides, P., Goulas, A., Yakinthos, K.: The hcuav project: Electronics and software development for medium altitude remote sensing. In: IEEE Int. Symp. on Safety, Security, and Rescue Robotics. pp. 1–5 (2014)
2. Amanatiadis, A., Bampis, L., Gasteratos, A.: Accelerating single-image super-resolution polynomial regression in mobile devices. IEEE Transactions on Consumer Electronics 61(1), 63–71 (2015)
3. Choi, K., Lee, I.: A uav-based close-range rapid aerial monitoring system for emergency responses. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci 38, 247–252 (2011)
4. Cuda™: Nvidia corp., `http://www.nvidia.com/object/cuda_home_new.html`
5. Karakasis, E.G., Bampis, L., Amanatiadis, A., Gasteratos, A., Tsalides, P.: Digital elevation model fusion using spectral methods. In: IEEE International Conference on Imaging Systems and Techniques. pp. 340–345 (2014)
6. Küng, O., Strecha, C., Beyeler, A., Zufferey, J.C., Floreano, D., Fua, P., Gervaix, F.: The accuracy of automatic photogrammetric techniques on ultra-light uav imagery. In: UAV-g 2011-Unmanned Aerial Vehicle in Geomatics (2011)
7. The lidar DEM data website, `http://b5m.gipuzkoa.net/`
8. Paull, L., Thibault, C., Nagaty, A., Seto, M., Li, H.: Sensor-driven area coverage for an autonomous fixed-wing unmanned aerial vehicle. IEEE Transactions on Cybernetics 44(9), 1605–1618 (2014)
9. Qi, H., Moore, J.B.: Direct kalman filtering approach for gps/ins integration. IEEE Transactions on Aerospace and Electronic Systems 38(2), 687–693 (2002)
10. Remondino, F., Barazzetti, L., Nex, F., Scaioni, M., Sarazzi, D.: Uav photogrammetry for mapping and 3d modeling–current status and future perspectives. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 38(1), C22 (2011)
11. Thomas, U., Kurz, F., Rosenbaum, D., Mueller, R., Reinartz, P.: Gpu-based orthorectification of digital airborne camera images in real time. In: Proceedings of the XXI ISPRS Congress (2008)
12. Wang, Y., Fevig, R., Schultz, R.R.: Super-resolution mosaicking of uav surveillance video. In: IEEE International Conference on Image Processing. pp. 345–348 (2008)
13. Xiang, H., Tian, L.: Method for automatic georeferencing aerial remote sensing (rs) images from an unmanned aerial vehicle (uav) platform. Biosystems Engineering 108(2), 104–113 (2011)