Real-time georeferencing for an airborne hyperspectral imaging system

Thomas Opsahl[*a], Trym V. Haavardsholm[a], Ingebrigt Winjum[a]

[a]Norwegian Defence Research Establishment (FFI), Instituttveien 20, 2027 Kjeller, Norway

ABSTRACT

The paper describes the georeferencing part of an airborne hyperspectral imaging system based on pushbroom scanning. Using ray-tracing methods from computer graphics and a highly efficient representation of the digital elevation model (DEM), georeferencing of high resolution pushbroom images runs in real time by a large margin. By adapting the georeferencing to match the DEM resolution, the camera field of view and the flight altitude, the method has potential to provide real time georeferencing, even for HD video on a high resolution DEM when a graphics processing unit (GPU) is used for processing.

Keywords: Real-time, georeferencing, ray tracing, pushbroom, orthorectification, hyperspectral, GPU

1. INTRODUCTION

Hyperspectral imaging has several applications where real-time image processing is desirable. For airborne systems, this presents challenges related to onboard processing, including georeferencing of the images. The Norwegian Defence Research Establishment (FFI) has developed an airborne hyperspectral target detection demonstrator system for real-time hyperspectral target detection[1]. Like most airborne hyperspectral sensors, our system is based on pushbroom scanning, where each scan line, or frame, must be individually georeferenced.

Georeferencing is based on accurate navigation using GPS in combination with an inertial sensor. From the navigation data, it is possible to estimate the camera position and orientation for individual frames. In combination with a geometrical camera model, this allows each pixel to be represented as a mathematical ray which models the line of sight for the corresponding detector element. The geographical position of the pixel can be estimated as the location where this ray intersects a digital elevation model (DEM) of the terrain. For real-time applications, the system must georeference pixels at least as fast as they are produced by the cameras. Several different approaches to this intersection problem can be found in the literature, exemplified by the iterative ray-tracing methods described by Schläpfer[2] and Müller[3]. However, these methods are not well suited for real-time applications.

To achieve real-time performance, the intersection problem must be solved very efficiently. Within the field of computer graphics, much research has gone into solving a similar problem. In order to display realistic images of a virtual 3D model, rays are cast from a virtual camera, and by simulating how light interacts with different surfaces, the color of the pixel corresponding to each ray gets determined. In this paper we outline a method for real-time georeferencing based on highly efficient ray-tracing algorithms used in computer graphics. We then describe the performance obtained from two different implementations, and give example results.

2. GEOREFERENCING BY EFFICIENT RAY TRACING

In this section we briefly outline the georeferencing algorithm. The rays representing the line of sight for each pixel can be determined using the position and orientation estimates from the navigation solution and a geometrical camera model. The terrain can be represented as a 3D triangle mesh by a Delaunay triangulation of the DEM vertices. At the core of the

[*] Thomas-Olsvik.Opsahl@ffi.no

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069

ray-tracing algorithm is an efficient ray-triangle intersection test. This test decides if a given ray intersects a given triangle or not. One way of doing this is to first calculate the point of intersection between the ray and the plane defined by the triangle and afterwards check if this intersection is enclosed by the triangle's edges. A brute force solution to the intersection problem would be to loop all rays over all triangles. In order to limit the number of intersection tests per ray, an accelerating data structure is introduced[4], which basically is a regular subdivision of the DEM's bounding box into volume elements (VE). This dramatically improves performance beyond the brute force approach, since the set of triangles that a ray can intersect is restricted to those occupying the VEs that are traversed by the ray. Moreover, the regular structure makes it computationally very cheap to step a ray from VE to VE.

The DEM and the rays must be represented in a common Cartesian coordinate system. When considering a local section of the DEM, the north-east-down (NED) coordinate system at its center is a convenient candidate. Standardized DEMs tend to have a regular structure in one of the common map projections, but this regularity will be lost when represented in the local NED coordinates. In combination with a regular spatial subdivision, this will cause many triangles to occupy more than one VE. For a ray traversing several VEs, this leads to a substantial amount of unnecessary ray-triangle intersection tests, as well as a more complex data structure. This can be avoided by resampling the DEM onto a regular grid in the local horizontal plane and applying the accelerating data structure to this resampled DEM instead. If the spatial resolution of the grid is similar to that of the original DEM, then the local section can be resampled without significantly altering the resolution, as illustrated in Figure 1. Moreover, the resampling can be performed offline, and consequently does not have to impact the real-time performance of the georeferencing method.

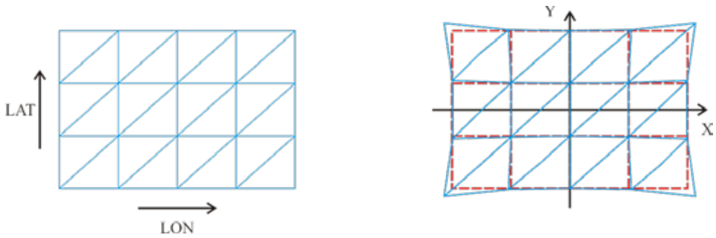

Figure 1. A regular grid in geodetic coordinates is only nearly regular in a local NED coordinate system. By resampling a local section of the DEM onto a suitable grid, the resampled DEM will not differ significantly from the original.

By letting every VE be vertically bounded by the DEM's bounding box, and subdividing horizontally according to the regular structure of the resampled DEM, the accelerating data structure ensures that every triangle occupies exactly one VE. Hence the combination of a regularly resampled DEM and a regular spatial subdivision lets each ray traverse efficiently through the grid of VEs, while greatly restricting the number of ray-triangle intersection tests. Moreover every intersection test is simplified by the fact that the triangle vertices are implicitly described by the structure.

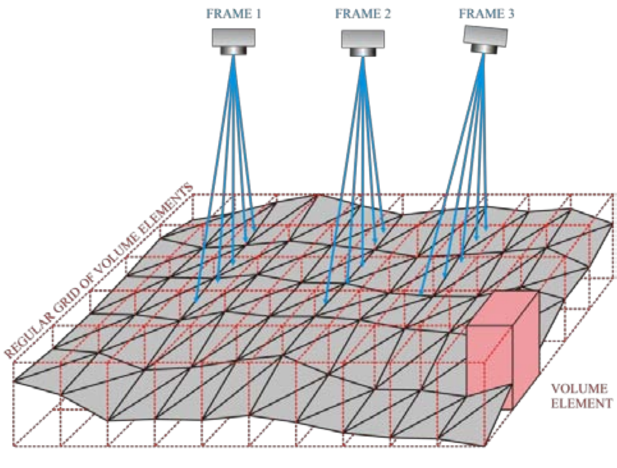

Figure 2. The regular subdivision can be chosen to match the regular grid used for resampling of the DEM. With this particular choice, every volume element contains 2 triangles where the vertices are aligned with the edges of the volume elements. Based on this regular structure, the proposed method solves the ray-DEM intersection problem very efficiently.

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069

Based on the geometry in the ray-DEM intersection problem, the computational workload of georeferencing can be further reduced. Due to the typically low spatial resolution of the DEM relative to the image, it is sufficient to georeference a subset of pixels for every frame. The geographical position of the remaining pixels can be accurately determined by interpolation. Due to the motion of the platform, subsampling along track should be avoided. For a nadir-oriented camera with a field of view θ, positioned at the height H over a flat regular DEM with spatial resolution L, one can easily show that $\left\lceil 1+2\frac{H}{L}\tan\frac{\theta}{2} \right\rceil$ is the minimum number of cross-track pixels required to achieve the same resolution in the georeference. For a high resolution camera, the workload can therefore be dramatically reduced by georeferencing only as many pixels as required by the resolution of the DEM. In the following, we demonstrate that the procedure outlined in this section can provide very fast georeferencing.

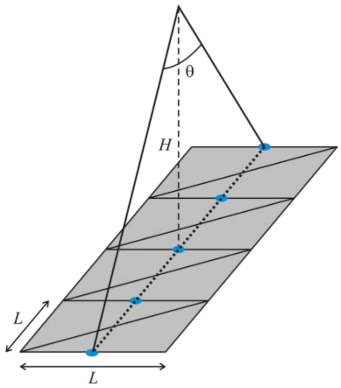

Figure 3. By calculating the ray-DEM intersection for a selection of pixels per frame (in blue), while determining the rest by interpolation, one can reduce the computational workload by a substantial amount without significantly degrading the georeference. Note that this scheme reflects the resolution of the DEM and not the resolution of the triangulation.

3. GEOREFERENCING IN THE SYSTEM

In FFI's demonstrator system, the main camera is a HySpex VNIR-1600 hyperspectral (HS) camera with 1600 pixels across track. The second camera is a high resolution (HR) Dalsa Piranha2 panchromatic camera with 8192 pixels across track and a 180 mm apochromatic objective lens. Both are pushbroom cameras with approximately 17 degrees field of view. The cameras are installed in a Cessna 172 aircraft. At a nominal altitude of 1000 meters above ground level, the cameras cover a roughly 300 meter wide swath with a nadir view. Given that the two cameras scan the scene at about 100 and 1000 lines per second, they produce more than 8,3 Mpixels/s in total. For navigation, the system employs a dual frequency GPS receiver and a navigation-grade inertial measurement unit (IMU).

A custom made digital synchronization unit time stamps synchronization pulses for individual frames from the cameras as well as the navigation data from the GPS and the IMU. This relates individual scan lines to GPS time. The navigation data are Kalman filtered by the real-time navigation software NavP developed at FFI. NavP is a real-time version of the navigation software package NavLab[5].

The streams of navigation data and time stamps for the camera frames are read over TCP/IP by a program running in MATLAB (a temporary prototype). Based on geometric camera models, individual pixels for each frame are represented as rays in a local NED coordinate system together with a resampled local section of the DEM. Using the described ray-tracing algorithm, the ray-DEM intersections are efficiently calculated and transformed into geodetic coordinates and streamed to the visualization software HyView, which is also developed at FFI. HyView then orthorectifies each image by draping it as a texture over a 3D surface determined by the corresponding georeference.

Using the data flow described above, the target detection demonstrator system is able to continuously display orthorectified image blocks with a few seconds latency. For real-time georeferencing, the terrain is modeled by DTED2 which has the spatial resolution of 1 arc-second, or roughly 30 meters. Hence for ideal conditions it is sufficient to consider $2\frac{H}{L}\tan\frac{\theta}{2} = 2\cdot\frac{1000\text{m}}{30\text{m}}\cdot\tan\frac{17°}{2} \approx 10$ pixels per frame for georeferencing. To make sure that the across track spatial resolution of the georeference is fine enough to tolerate the typical platform motion, 20 regularly distributed pixels per

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069

frame are considered. Hence, out of the roughly 8.3 Mpixels/s produced by the two cameras, only 22 kpixels/s are georeferenced by ray tracing, while the rest is implicitly georeferenced by interpolation in the visualization on the GPU. This subsampling effectively reduces the workload of georeferencing by 99.7% with negligible loss in accuracy. Figure 4 shows an example result from the real-time demonstrator system, where a pushbroom image is rectified based on a georeference with 20 pixels per frame.

Figure 4. The leftmost image is an RGB representation of a raw HS image taken during a sharp left banking turn. To the right, the same image is shown rectified on top of a topographic map[6] in HyView. Zoom images from the HS and HR imagery have been added to illustrate the quality of the rectification.

4. PERFORMANCE

To demonstrate the performance of the ray-tracing method, it was tested on data from a simulated straight and level flight 1000 meters above a flat DEM at a typical cruise speed. To make it relevant for the target detection system, the camera was chosen to be nadir oriented and to have 17 degrees field of view. By making the DEM's bounding box 1000 meters tall, all rays were forced to move through the maximal amount of VEs before the valid intersections were determined. For different combinations of DEM resolutions and number of rays, the real-time performance of the ray-DEM intersection method was computed. The results do not reflect the time spent on coordinate transformations or resampling, since their contribution to the overall performance is less significant.

4.1 Performance for the target detection demonstrator system

The test was performed both with and without subsampling. For the test with subsampling, a subset of $\left\lceil 4\frac{H}{L}\tan\frac{\theta}{2}\right\rceil$ regularly distributed pixels was considered for each frame, and the processing was performed on blocks of rays corresponding to 1 second of data. With spatial DEM resolutions of 100, 30, 10 and 5 meters, the chosen subsampling corresponds to 6, 20, 60 and 120 pixels. For the test without subsampling, the processing was performed on 0.01 second blocks at a time, due to the large number of rays. The test was performed using the code from the demonstrator system running MATLAB R2010a on a PC with a 2.66GHz Quad Core Intel Xeon CPU and 12GB RAM.

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069

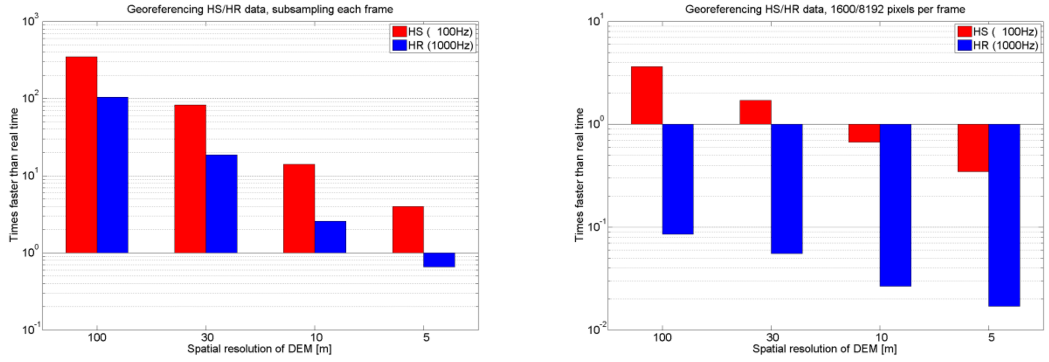

Figure 5. These results show the performance of the ray-tracing solution to the ray-DEM intersection problem, with and without subsampling of pixels. It is clear that demonstrator system, which is based on a DEM with roughly 30 meters spatial resolution, can achieve real-time performance when subsampling each frame.

The results in Figure 5 show that the presented ray-tracing method, with subsampling of each frame, provides real-time performance for the test flight using a DEM with 10 meter spatial resolution. For the demonstrator system, using DTED2 with roughly 30 meters spatial resolution, the test results show that when subsampling is performed, HS and HR data are georeferenced roughly 80 and 19 times faster than real-time. This corresponds to approximately 12.8 Mpixels/s and 155.6 Mpixels/s respectively. It is also clear that more computational power is needed if high resolution DEMs are to be used.

4.2 Performance using CUDA

In order to investigate the potential of the georeferencing method, a prototype version using CUDA[7,8] in C++ has been implemented to exploit the computational power available in graphics processing units (GPU). Making use of parallel processing techniques, several bundles of rays can be traced simultaneously on the GPU. Due to the nature of parallel processing on the GPU and memory transfers between the GPU and the CPU, the performance relies heavily on the number of rays that can be processed simultaneously. The performance drops when too few rays are processed at a time, in which case the GPU architecture is not being efficiently used. Thus by allowing some latency in the georeferencing process, performance can be increased. The results in Figure 6 show the performance using CUDA without subsampling on a NVIDA GeForce GTX 580.

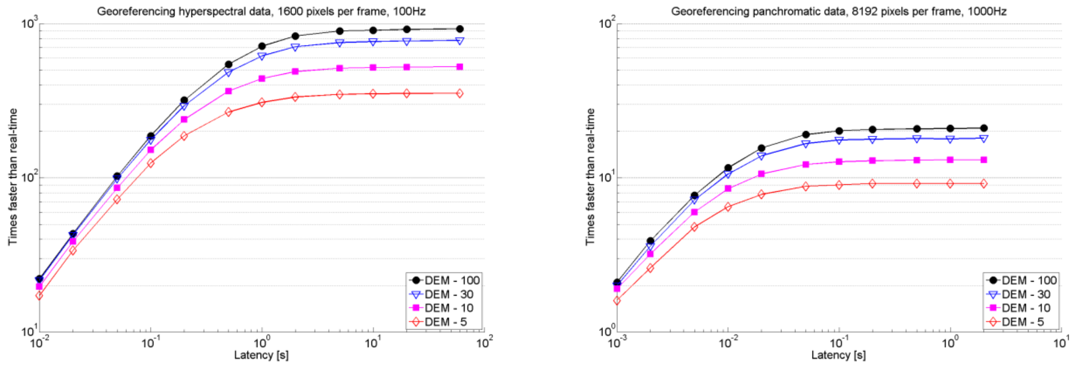

Figure 6. These graphs illustrate that by using a GPU it is possible to georeference the images produced in the target detection system in real time, without subsampling. Even when the terrain is modeled by a DEM with 5 meter spatial resolution, the panchromatic data can be georeferenced 9 times faster than real-time with a latency of 0.1s.

These results show that the presented ray-tracing approach to georeferencing has potential for real-time performance even for very high-resolution DEMs. The GPU provides maximum efficiency when processing blocks with about 200 000 rays. By simultaneously processing rays from both cameras, it seems possible to georeference all pixels almost 10 times faster than real-time with a latency of less than 0.1 second. With subsampling the processing becomes a further factor of 10 times faster without reaching maximum efficiency for 0.1 second latency.

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069

The proposed method is also relevant for other sensors than pushbroom cameras. Figure 7 shows the performance for a simulation using a 60 Hz HD video camera with 1920x1080 pixels and a 17° wide field of view. For this particular test no subsampling was performed. Due to the huge amount of pixels, even a single frame was enough to make efficient use of the GPU, resulting in minimal latency.

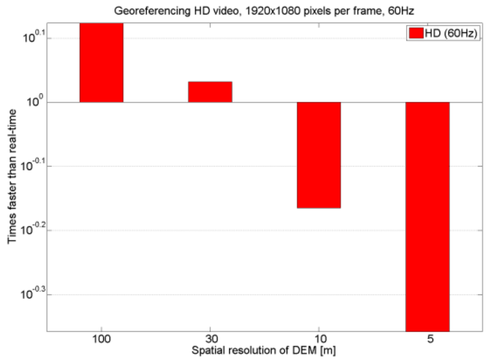

Figure 7. When using a GPU for processing, the full resolution HD video can be georeferenced in real time using a DTED2 DEM.

The results indicate that even without subsampling of each frame, real-time georeferencing of 60Hz HD video is possible to achieve with the proposed ray-tracing method. If each frame can be subsampled to some extent, the performance will increase dramatically.

5. SUMMARY

The georeferencing part of FFI's target detection demonstrator system has been presented. Real-time georeferencing is achieved due to the use of ray-tracing methods from computer graphics, a highly efficient representation of the terrain and a scheme for reducing the problem by adapting to the spatial resolution of the terrain model. Test results show that the demonstrator system performs in real-time when DTED2 is used to model the terrain. By exploiting the computational power of the GPU, the method has the potential for real-time performance even for very high resolution terrain models and video.

REFERENCES

[1] Skauli, T., Haavardsholm, T. V., Kaasen, I., Arisholm, G., Kavara, A., Opsahl, T. O., Skaugen, A., "An airborne real-time hyperspectral target detection system", Proc. SPIE 7695, 7650A, (2010).

[2] Shläpfer, D. and Richter, R., "Geo-atmospheric processing of airborne imaging spectrometry data. Part 1: Parametric orthorectificatin", International Journal of Remote Sensing, Vol. 23, 2609-2630, (2002).

[3] Müller, R., Lehner, M., Müller, R., Reinartz, P. and Schröder, M., "A program for direct georeferencing of airborne and spaceborne line scanner images", Proc. ISPRS Commission I Symposium: Integrating Remote Sensing at the Global, Regional and Local Scale, Vol. XXXIV, Part 1, (2002).

[4] Ericson, C., [Real-Time Collision Detection], Morgan Kaufmann Publishers, (2005).

[5] Gade, K., "NAVLAB, a Generic Simulation and Post-processing Tool for Navigation", European Journal of Navigation Vol. 2, no. 4, (2004).

[6] http://www.norgeskart.no.

[7] NVIDIA CUDA, http://developer.nvidia.com/object/cuda.html.

[8] Haavardsholm, T.V., Winjum I., Opsahl, T., "Real-time direct georeferencing of images from airborne line scan cameras", Poster, GPU Technology Conference 2010, 20-23 September 2010, San Jose.

Dette er en postprint-versjon/This is a postprint version.
DOI til publisert versjon/DOI to published version: 10.1117/12.885069