Exploring the Registration of Remote Sensing Images using HSI–KAZE in Graphical Units

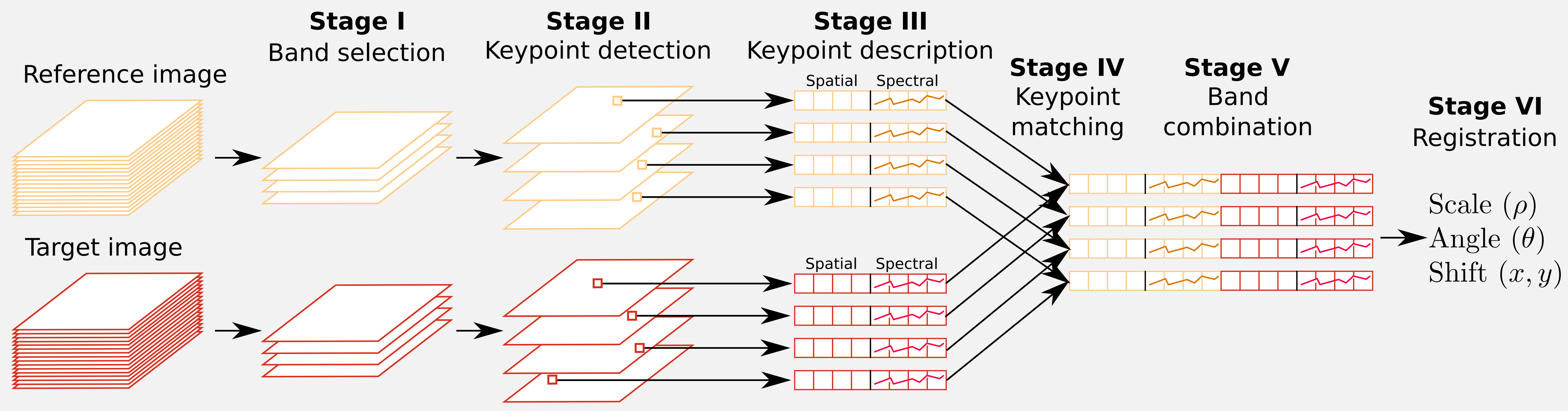
Álvaro Ordóñez [1], Dora B. Heras [1*], and Francisco Argüello [2]

[1] Centro Singular de Investigación en Tecnoloxías Intelixentes, Universidade de Santiago de Compostela
[2] Departamento de Electrónica e Computación, Universidade de Santiago de Compostela
[*] E-mail: dora.blanco@usc.es

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
CiTIUS Centro Singular de Investigación en Tecnoloxías Intelixentes
19th International Conference Computational and Mathematical Methods in Science and Engineering 2019 CMMSE

Registration of hyperspectral remote sensing images is a common task in many image processing applications such as land use classification, environmental monitoring and change detection. The images to be registered present differences as a consequence of being obtained from different points of view, differences in the number of spectral bands captured by the sensors, in illumination and intensity, and also changes in the objects present in the images, among others. Feature–based methods as HSI–KAZE are more efficient at registering than area-based methods when the images are very rich in geometrical details, as it is the case for remote sensing images. But they present, nevertheless, the problem of being computationally more costly because the number of distinctive points to be calculated for these images is high. HSI–KAZE is a method to register hyperspectral remote sensing images based on KAZE features but considering the spectral information. In this work, a robust and efficient implementation of this method on programmable GPUs is presented.

HSI–KAZE on GPU



Algorithm 1 HSI–KAZE pseudocode.

Input: Hyperspectral reference image $\mathbf{I}_1$ and hyperspectral target image $\mathbf{I}_2$ with N_T bands.
Output: Scale factor ρ, rotation angle θ, and translation (x, y).

Perform feature selection over both images $\rightarrow \mathbf{B}_1$ and $\mathbf{B}_2$ ▷ Band selection

for each band b **in** images $\mathbf{B}_1$ and $\mathbf{B}_2$ **do**
  Extract keypoints of $\mathbf{B}_1^b \rightarrow \mathbf{P}_1^b$ ▷ Keypoint detection
  Extract keypoints of $\mathbf{B}_2^b \rightarrow \mathbf{P}_2^b$ ▷ Keypoint detection
  Calculate the M–SURF descriptor of each keypoint in $\mathbf{P}_1^b$ and append the spectral signature $\rightarrow \mathbf{K}_1^b$ ▷ Keypoint description
  Calculate the M–SURF descriptor of each keypoint in $\mathbf{P}_2^b$ and append the spectral signature $\rightarrow \mathbf{K}_2^b$ ▷ Keypoint description
  Match keypoints in $\mathbf{K}_1^b$ and $\mathbf{K}_2^b \rightarrow \mathbf{M}_b$ ▷ Keypoint matching
end for

Combine all the matched keypoints $\mathbf{M}_b \rightarrow \mathbf{M}$ ▷ Band combination
Perform an exhaustive search to recover the registration parameters $\rightarrow \rho, \theta, (x, y)$ ▷ Registration


(a) (b) (c)

Figure: Example of registration considered in this work: a) Reference image (size 1096 × 715), b) Target image, and c) Result of the registration process showing the correctly registered superposition of the reference and target registered image (scale 23.0× and rotation angle 60°).

Keypoint detection and description on GPU

- Acronyms:
  - $<\ >$: Function executed in the GPU (kernel).
  - GM: Target function data is allocated in Global Memory.
  - SM: Target function data is allocated in Shared Memory.

Algorithm 2 Pseudocode for the keypoint detection and description stages.

Input: Set of selected bands $\mathbf{B}_1$ and $\mathbf{B}_2$.
Output: A set of keypoints $\mathbf{K}$ for each selected band of both images.
Parameters: Number of sublevels N_{sub}.

Keypoint detection
Calculate the optimal number of octaves N_{oct} according to the spatial size of the images
$<$ Upsample the images to obtain images whose size is divisible by the number of octaves N_{oct} $>$ ▷ GM
for each band b **in** images $\mathbf{B}_1$ and $\mathbf{B}_2$ **do**
  $<$ Upsample the band by a factor of 2 using bilinear interpolation $>$ ▷ GM
  stage Build the pyramidal scale space
    $<$ Smooth the upsampled band using a Gaussian filter $>$ ▷ GM+SM
    $<$ Compute the Scharr derivatives $>$ ▷ GM+SM
    $<$ Compute the contrast factor k from the gradient histogram of the smoothed band $>$ ▷ GM+SM
    sub-stage Build the pyramid using FED scheme
      for $o \leftarrow 1, N_{oct}$ **do**
        $<$ Subsample the last sublevel image by a factor of 2 $>$ ▷ GM+SM
        for $s \leftarrow 1, N_{sub}$ **do**
          $<$ Smooth using a Gaussian filter $>$ ▷ GM+SM
          $<$ Compute the Scharr derivatives $>$ ▷ GM+SM
          $<$ Compute the conductivity g $>$) ▷ GM
          $<$ Discretized the nonlinear diffusion equation using the FED scheme $>$ ▷ GM+SM
        end for
      end for
    end sub-stage
  end stage

  stage Locate the keypoints in the scale space
    $<$ Compute the determinant of the Hessian matrix $>$ ▷ GM+SM
    $<$ Detect keypoints by searching for points that are the maxima of their neighbourhood $\rightarrow \mathbf{K}_1^b, \mathbf{K}_2^b$ $>$ ▷ GM
    $<$ Refine the position and the scale of each keypoint $>$ ▷ GM+SM
  end stage

Keypoint description
  for each keypoint **in** $\mathbf{K}_1^b$ and $\mathbf{K}_2^b$ **do**
    $<$ Calculate the main orientation $>$ ▷ GM
    $<$ Compute the M–SURF descriptor $>$ ▷ GM
    Append the spectral signature
  end for
end for

Keypoint matching on GPU

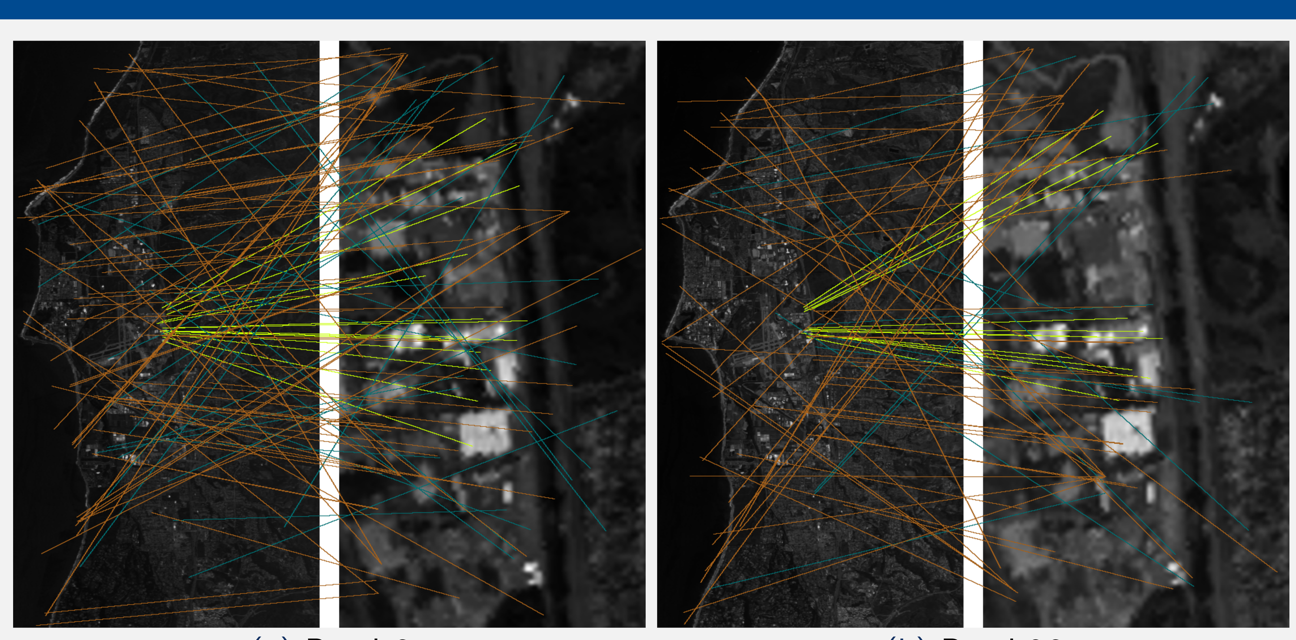

(a) Band 6 (b) Band 26

Figure: Matched keypoints detected in two of the eight selected bands belonging to the Santa Barbara Front scene with scale 9.5×: matches discarded after considering spectral information (brown), incorrect matches (blue), correct matches (yellow), and correct matches used in registration (green).

Table: Number of matches obtained for Santa Barbara Front in the eight selected bands.

Number of matches	44550
Number of matches after spectral discarding	44490
Number of matches after removing repeated matches	43537
Number of correct matches	20741
Number of incorrect matches	22796

Registration results

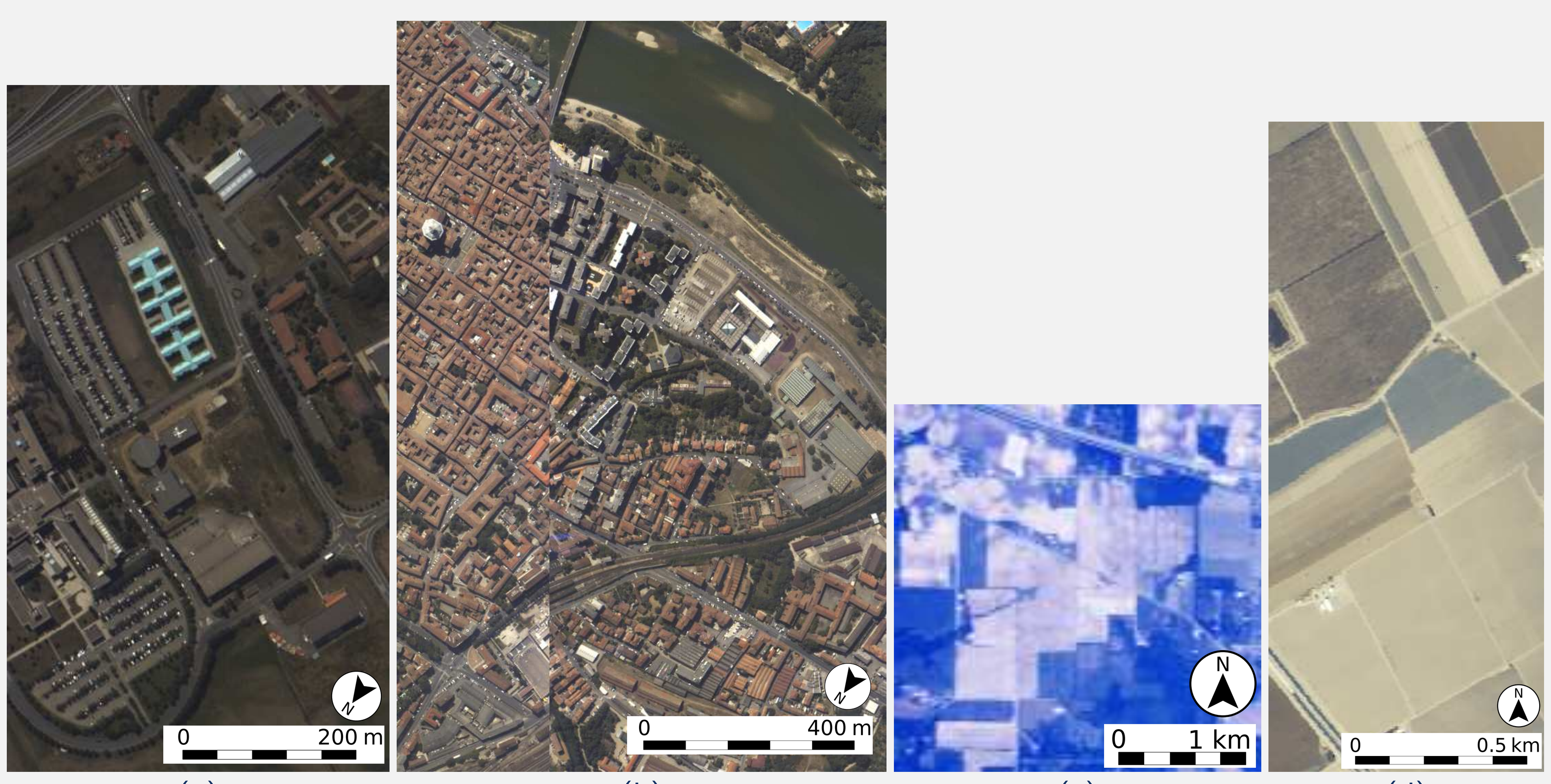
Table: Successfully registered cases for each scene and hyperspectral registration method. × means "scaling factor" and for each scaling 72 angles are considered. HYFM is not based on feature extraction.

Scene	HYFM	HSI–KAZE
Pavia University	1/4× to 5.5 × (13)	1/11× to 13.0 × (35)
Pavia Centre	1/5× to 7.5 × (18)	1/16× to 24.0 × (62)
Indian Pines	1/2× to 4.0 × (8)	1/4× to 5.5 × (13)
Salinas	1/2× to 4.5 × (9)	1/7× to 6.0 × (17)
Number of scalings	(11.43)	(30.71)

Experimental conditions

Table: Sensor, size, number of spectral bands, resolution (m/pixel), and location of the test hyperspectral images

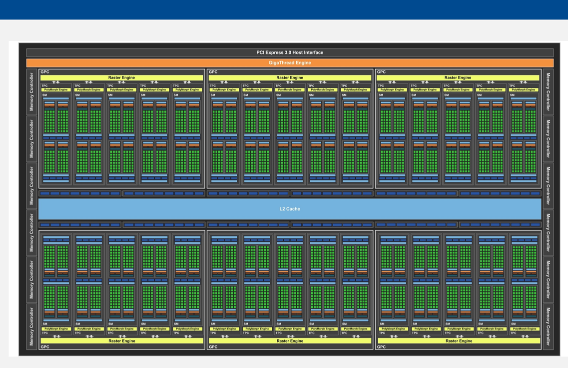
Image	Sensor	Size	Bands	Spatial Resolution
Pavia University	ROSIS–03	610 × 340	103	1.3
Pavia Centre	ROSIS–03	1096 × 715	102	1.3
Indian Pines	AVIRIS	145 × 145	220	20.0
Salinas	AVIRIS	512 × 217	204	3.7



(a) (b) (c) (d)

Figure: Remote sensing hyperspectral images: a) Pavia University, b) Pavia Centre, c) Indian Pines, and d) Salinas.

GPU optimization strategies



- Memory hierarchy in the GP102 Tesla architecture:
  - 96 KB/SM of shared memory.
  - 48 KB/SM of L1/texture cache
  - 3072 KB of L2 cache.
- A set of strategies to reduce the computational time have been applied:
  1. Reduce data transfers among CPU and GPU memories
  2. Reuse data in shared memory.
  3. Search for the best kernel configuration to reduce the execution time and maximize the GPU occupancy.
  4. The use of atomic operations prevents the race conditions among threads.
  5. Efficient computation using optimized CUDA libraries.

Experimental results

- Intel Xeon E5–2623v4 CPU at 2.60 GHz.
- 128 GB of RAM
- Ubuntu 16.04.6 LTS.
- NVIDIA P40 (GP102).
  - 30 SMs, 128 CUDA cores/SM.
  - 24 GB of GM.
  - 1303 MHz of base clock.

Table: CPU and P40 GPU computation times for each scene.

	CPU	GPU	Speedup
Pavia University	71.75s	12.11*s*	5.92×
Pavia Centre	508.61s	45.30*s*	11.23×
Indian Pines	5.40s	1.90*s*	2.84×
Salinas	23.63s	5.23*s*	4.52×

Table: Detailed times for each stage projected onto GPU for Pavia Centre scene.

	CPU	GPU	Speedup
Band selection	10.87s	0.24*s*	45.29×
Keypoint detection	401.41s	12.75*s*	31.48×
Keypoint description	73.71s	13.83*s*	5.33×
Keypoint matching and band combination	22.37s	17.55*s*	1.27×

Conclusions

- An efficient CUDA GPU implementation of HSI–KAZE that registers hyperspectral images is presented.
- HSI–KAZE exploits the spectral information available in the images.
- The algorithm performs successful registration for scale factors of up to 24.0× for all the rotation angles and translations.
- Speedups of up to 11.3× for real remotes sensing images are achieved.
- Execution time reduction from around 9 minutes to less than 1 minute for the biggest test set image.

References

[1] Álvaro Ordóñez, F. Argüello, and D. B. Heras, "Alignment of hyperspectral images using KAZE features," *Remote Sensing*, vol. 10, no. 5, 2018. [Online]. Available: http://dx.doi.org/10.3390/rs10050756

[2] K. Yang, A. Pan, Y. Yang, S. Zhang, S. H. Ong, and H. Tang, "Remote sensing image registration using multiple image features," *Remote Sensing*, vol. 9, no. 6, 2017. [Online]. Available: https://www.mdpi.com/2072-4292/9/6/581

[3] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "KAZE features," in *European Conference on Computer Vision*. Springer, 2012, pp. 214–227.

[4] P. F. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Patt. Anal. Mach. Intell*, vol. 34, no. 7, pp. 1281–1298, 2011.