A Scale-invariant Change Detection Method for Land Use/Cover Change Research

*Jin Xing, Renee Sieber, and Terrence Caelli*

1 Department of Geography, McGill University, Montreal, Canada
Email: jin.xing@mail.mcgill.ca; renee.sieber@mcgill.ca

2 Department of Electrical and Electronic Engineering, The University of Melbourne, Melbourne, Australia
Email: tcaelli@unimelb.edu.au

**Abstract**

Land Use/Cover Change (LUCC) detection relies increasingly on comparing remote sensing images with different spatial and spectral scales. Based on scale-invariant image analysis algorithms in computer vision, we propose a scale-invariant LUCC detection method to identify changes from scale heterogeneous images. This method is composed of an entropy-based spatial decomposition, two scale-invariant feature extraction methods, Maximally Stable Extremal Region (MSER) and Scale-Invariant Feature Transformation (SIFT) algorithms, a spatial regression voting method to integrate MSER and SIFT results, a Markov Random Field-based smoothing method, and a support vector machine classification method to assign LUCC labels. We test the scale invariance of our new method with a LUCC case study in Montreal, Canada, 2005-2012. We found that the scale-invariant LUCC detection method provides similar accuracy compared with the resampling-based approach but this method avoids the LUCC distortion incurred by resampling.

**Keywords:** Land Use/Cover Change Detection; Scale Variance; Scale-invariant Feature Transformation; Maximally Stable Extremal Region; Hadoop; Cloud Computing.
1. Introduction

Big data provides us with numerous new sources of data for Land Use/Cover Change (LUCC) but it causes problems related big data’s large volume, complex variety, increasing velocity, and challenging verification (Miller and Goodchild, 2015). We now have finer spatio-temporal resolutions of LUCC data, with greater variety in terms of spectra and sensing platforms (Hansen et al., 2013). Larger, faster, and diverse data offers significant potential for LUCC but it quickly exceeds the data handling capacity and capability of existing LUCC algorithms. Among the four “Vs” of big data, volume is predominant focus in LUCC research (e.g., Hampton et al., 2013), although velocity also has attracted interest (Gil-Yepes et al., 2016; Wu, Zhang, and Du, 2017; Wu et al., 2017). Our paper emphasizes the variety and specifically the various scales that are now available (i.e., different spatial, spectral, and temporal granularities and extents) (Goodchild, 2011). Because LUCC uses two or more datasets to identify changes, big data introduces potential problems in scale variance (Woodcock and Strahler, 1987).

If scales vary, one usually interpolates or re-samples one or more datasets to homogenize spatial granularities (i.e., resolutions) and extents to co-register the images for LUCC detection (e.g., Zhang et al., 2016). These spatial scaling operations can cause various problems like the generation of erroneous artefacts (Kwok and Sun, 1993), loss of information (Sheikh and Bovik, 2006), and distortion of geographic entities (Prashanth et al., 2009). As a result of these spatial scaling operations, LUCC accuracy can be significantly degraded by scale variance (Olofsson et al., 2014) particularly if we wish to take advantage of the high resolution characteristic of big data.

To avoid the drawbacks of using spatial interpolation or re-sampling techniques, research scientists have investigated novel solutions to handle the challenge of scale
variance. For example, Chen et al. (2012) clustered pixels into image objects prior to comparison and then compared the geo-registered objects from datasets at two different scales. Singh (1989) bypassed the comparison of image pixels and explored a post-classification method to extract LUCC by comparing the class label maps. Xiao et al. (2016) combined pixel-based method and object-based approach together to investigate urban LUCC with high-resolution imagery datasets. All these approaches assume that the scale variance in any LUCC would be minor and that geo-registration would be sufficient to compare image objects. Big data does not make these assumptions by creating new multi-scale challenges for the study of LUCC.

Computer vision algorithms have been explored to tackle challenges introduced by different kinds of variance (Radke et al., 2005). These algorithms are interesting because they focus on differentiating objects within datasets and do not rely on geo-registration because the objects may be moving image to image. Scale-invariant computer vision algorithms exploit scale by artificially deriving multiple images, each at a different resolution, from a single image. They then extract the stable "scale-invariant" features from these derived images. An example of the utility of scale-invariant computer vision for LUCC can be found in Dellinger et al. (2014). They proposed using the Scale-invariant Feature Transformation (SIFT) (Lowe, 2004) to handle images from diverse sensing platforms. Pham, Mercier, and Michel (2016) employed SIFT to study LUCC before and after a volcanic eruption. They used SIFT to reduce variations in each image from illumination, color, or view angle differences. Authors of these two papers compared images at the same resolution. Ye et al. (2014) utilized another computer vision algorithm, Speed Up Robust Features (SURF), with images of different resolutions
for LUCC. However, they resampled the images to create scale homogeneity and then extracted the changed regions. We want to exploit computer visions algorithms proven to identify scale-invariant features for LUCC: to detect scale-invariant changes across multiple remote sensed (RS) images of different resolutions.

Our scale-invariant LUCC detection method integrates spatial decomposition, image feature comparisons that are derived from computer vision, change map smoothing, and LUCC labelling. We will show that: (1) LUCC can be extracted by comparing scale-invariant image features directly without spatial interpolation or re-sampling methods; (2) discrimination of scale-invariant image features can be enhanced by the integration of extent, shape, and spectral information for LUCC; and (3) high performance computing can provide significant support in the scale-invariant LUCC detection workflow.

The rest of this paper is organized as follows. Section 2 enumerates the benefits and challenges of scale-invariant algorithms derived from computer vision. Our scale-invariant LUCC detection method is introduced in Section 3, which is based on the integration of SIFT and the Maximally Stable Extremal Region (MSER). Section 4 is a case study in the Greater Montreal Area from 2006-2012, which evaluates our scale-invariant LUCC detection algorithm. This paper concludes in Section 5.

2. Handling Scale Variance with Computer Vision Algorithms
A large body of computer vision algorithms employ scale variance. One widely applied approach is feature detection, expressed in algorithms like SIFT, MSER, and the Gradient Location and Orientation Histogram. Image features are extracted that are stable across various granularities, which are derived as needed from a single original image (Witkin, 1984; Huo et al., 2008). Image fusion is another scale variance handling method in
computer vision, which merges relevant information from at least two images at different spectral and spatial granularities to achieve higher granularities (Li et al., 1995). For example, image fusion with multispectral IKONOS (4m, red, green, blue, and infra-red) and panchromatic (1m, greyscale) IKONOS images will generate a new image with 1m resolution and 4 bands of information.

Perona and Malik (1990) and others offered good examples of how computer vision studies differ from LUCC. Although they (ibid.) explored changes in image object boundaries at different spatial granularities, their study was conducted with everyday object extents (e.g., 1 mm at 1m²). LUCC works with larger extents and a broader range of granularities. Their study also was conducted with a single image but LUCC involves comparing images taken at different times. Their study considered changes in image object characteristics; however, LUCC functions at the image level and detects changes throughout the image extents. Ohn-Bar and Trivedi (2014) proposed a temporal interpolation algorithm to model the movement of human hand gestures. They studied a time span of deciseconds (100msec units or 0.1 of a second). The time span in LUCC datasets may be several years or decades. Non-linear temporal models (e.g., branch, cyclical, and isochronical models) may further complicate temporal scale variance (Jönsson and Eklundh 2004). Therefore, the scale variance in LUCC requires additional investigation before we can apply the computer vision algorithms.

2.1 Similarity of Land Use/Cover Entities
LUCC researchers have expressed considerable interest in SIFT. SIFT is an algorithm designed to detect, describe, and match key points across images. SIFT points are those points (pixels) that persist in the image regardless of various transformations. SIFT points are considered to be invariant to spatial granularity, rotation, affine distortion, translation,
and illumination differences. SIFT points are extracted from regions as the minima/maxima of Difference-of-Gaussians (Bundy and Wallen, 1984). Image matching, clustering, and pattern recognition are then performed by matching SIFT points.

Previous work has highlighted the deficiency of SIFT in distinguishing similar land use/cover entities, which mainly occur in the dense urban areas (Tuermer et al., 2013; Sirmacek and Unsalan, 2009). Entities such as those composed of cement (e.g., buildings and roads) can be so similar (Yang et al., 2003) that SIFT cannot adequately discriminate among them. In Figure 1 two images are carefully geo-registered but SIFT matching largely fails because of a lack of uniqueness in SIFT characteristics (e.g., for corners of roads and buildings).

**Figure 1.** SIFT comparison using 10 key points extracted from left (0.11m Montreal Montreal Metropolitan Community Orthophotos / Orthophotographies {MMCO} images recorded at downtown Montreal, 2005 {Communauté métropolitaine de Montréal, 2005}) and right (0.13m MMCO recorded at downtown Montreal, 2007 {Communauté métropolitaine de Montréal, 2007}) respectively. The two images are carefully geo-registered, but seven SIFT mismatches occur because urban structures are very similar to each other. Because SIFT uses 128-bit encoding, multiple points can have exactly the
same values (illustrated with the same color). 17 point pairs here are counted 10 SIFT key point pairs (illustrated by different colors).

2.2 Use of Shape Information
Regions are defined by geometric and topological connections among positions and features. Unsurprisingly, regions are sensitive to spatial granularity changes (Luo and Min, 2010). SIFT points (pixels) can be used to compare images directly and mark clusters of unmatched points as changed regions (Dellinger et al., 2014). Although change information can be represented by individual pixels, our approach considers change as a multi-scale collection of regions, which is more robust and more useful for LUCC. These multi-scale regions represent LUCC areas as clusters composed of different numbers of pixels across scale-heterogeneous remotely sensed images. Regions not only provide more information about LUCC (e.g., change boundaries and areas) but also should prove more resistant to noisy information.

As shown in Figure 2, numerous changed SIFT key points (red points) are caused by the noise or artefacts, such as shadows, vehicles, trees, and building decorations. Few of these changed points represent actual LUCC. To overcome noise or artefacts, we can combine SIFT with a computer vision algorithm like MSER (Matas et al., 2004). MSER extracts scale-invariant regions (clusters) and matches regions across various images. LUCC can take advantage of MSER’s use of shapes because unmatched MSERs represent changed regions.
SIFT matching-based change detection. (A) upper left corner tile (one ninth) of left image in Figure 1; Figure (B) is the corresponding upper left corner tile (one ninth) of right image in Figure 1; in (C) and (D), green points stand for the unchanged SIFT key points, and the red ones represent the changed SIFT key points. Matching is implemented with BoofCV (an open source Java library for computer vision and image analysis) using the same parameters as Figure 1.

2.3 Integration of Spectral Information

SIFT is designed for grey scale images and does not consider spectral information. Since LUCC imagery datasets are being generated by increasing numbers and variety of RS platforms, their spectral channels (bandwidth) can be diverse (Xu and Gong, 2007; Singh, 1989). Default SIFT only considers the image contrast intensity, so it may not handle the opportunities presented by this diversity.

Figure 1 illustrates the problems in only handling image contrast intensity. Roads are being matched up with buildings due to very similar intensity values in terms of grey scale. The original SIFT algorithm provides reasonable geometric distinction for object recognition but can prove inadequate for spectra (Abdel-Hakim and Farag, 2006). To address this limitation, the authors modified SIFT to use spectral information, which they
argued can enhance SIFT comparison performance for image object recognition. In Figure 3 the modified SIFT, or Color-SIFT (CSIFT), generates a fifteen percent improvement in matching CSIFT transforms the original image from RGB space into the spectral differential quotients, using the 3x3 Gaussian color transformation matrix (Fritz, Seifert, and Paletta, 2006). Then, the standard SIFT algorithm is applied to the spectral differential quotients to identify CSIFT key points and generate the CSIFT feature vectors. According to the authors (ibid.), SIFT with spectral information tends to extract more key points than the standard SIFT. CSIFT identifies more SIFT key points for LUCC detection and potentially lowers the risk of mismatching.

![Figure 3. Color SIFT matching. (A) and (B) are the spectral SIFT matching of Figure 2 (A) and (B), respectively. This follows Abdel-Hakim and Farag (2006)’s use of Gaussian color model for SIFT computation.](image)

3. A Scale-invariant LUCC Detection Method
To address the three main challenges of using scale-invariant image features in LUCC, we propose a scale-invariant LUCC detection method, which compares images of differing spatial scales (i.e., granularity and extent) without altering the original images via interpolation/re-sampling. The proposed method has five steps, each of which is described below. Because our goal is big data handling, the first step is to decompose the
image into small tiles using a spatial entropy formulation. Second, MSER extracts scale-invariant regions after which we perform a many-to-many matching operation to determine potential changed regions. Third, SIFT points are computed, and then combined with change-specific MSERs to detect changed regions where changes are not due to scale variance. These changed regions may contain noisy information (e.g., vehicles, trees, and shadows). Consequently, fourth, a change map smoothing method is used to reduce irrelevant information (e.g., shadows, trees, and vehicles). Finally, a classification algorithm labels the changes in the change map tiles. The workflow of the scale-invariant LUCC detection method is illustrated in Figure 4.
3.1 Data Decomposition
If we simply decompose two scale heterogeneous images into an even number of tiles then we may extract thousands of small MSERs at the fine granularity and a few large MSERs from the coarse image. This may generate thousands of changed regions that do not come from LUCC but from scale variance. We are inspired by Tan et al. (2007), who proposed an entropy-based equal histogram decomposition method for classification. Ours is an entropy-based equal-area splitting method to decompose large images into...
smaller tiles, so as to maximize detection of potential changed regions. The number of tiles will differ from image to image based on the degree of entropy in the image. This entropy-based image decomposition should improve matching MSER and SIFT by normalizing numbers of SIFT points and MSERs across tiles.

The spatial entropy method (Journel and Deutsch, 1993) is shown in equation (1). It extends the traditional entropy model, $H = -\sum_{i=1}^{n} P_i \log_2 P_i$, by normalizing the extent relative to resolution (Batty, 1974). This ensures that smaller areas with higher variance will be decomposed similarly to larger areas with lower variance. $P_i$ is the probability that the difference between two adjacent pixels is equal to $i$. Since the spatial granularity (resolution) is set, the extent will be adjusted to guarantee the same spatial entropy $E$ among the tiles.

$$E = (-\sum_{i=1}^{n} P_i \log_2 P_i) \ast (\log_2 \frac{\text{Extent}}{\text{Resolution}}) \quad (1)$$

In practice, it is difficult to achieve a perfect match of $E$ image to image, so we create a tolerance parameter called $\tau$, which is the maximum variance between the two entropy scores. Big data is often available at fine spatial granularities that are much smaller than the size of the landscape entities and any big data LUCC will invariably split some objects across multiple tiles (Xing, Sieber, and Kalacska, 2014).

3.2 MSER Extraction and Matching

In Step 2, MSER generates regions from each tile and then attempts to match them. We implement a color MSER extraction method (Forssén, 2007) that integrates spectral information into the feature extraction process. MSERs are generated from $n$ iterations of a “growing-and-merging” approach to segment an image tile into clusters of pixels (Zhu and Yuille, 1996). We systematically evaluate different thresholds in each iteration to test if the region boundaries remained stable (i.e., the boundary changes are smaller than the
maximum variation value-\textit{MaxVariation}) (Matas et al., 2004). In each iteration, the difference between the thresholds needs to be larger then a predefined value $m_{\text{min}}$ (we use the value from Forssén 2007). Regions are considered to be stable MSERs if they contain pixels larger than the minimum ($\text{MinArea}$). We further refine the matching potential with the RANdom SAmple Consensus (RANSAC) (Fischler and Bolles, 1981) algorithm, which is commonly used in combination with MSER (Cheng, et al., 2012). The parameters are usually tuned with training datasets or determined heuristically (Forssén, 2007).
Figure 5. MSER matching across image tiles. (A) The unchanged MSERs extracted from Image $X_1$, 0.11m MMCO image tile recorded in 2005 at downtown Montreal. Figure (B), (C), (D), and (E) are the four coarser granularity tiles with the highest MSER matching scores, using 0.13m MMCO recorded in 2007 (from Image $X_2$). Unchanged MSER “mask” is depicted with green boundaries. Other image tiles with MSER matching scores lower than the threshold ($\text{MinArea}$) are not considered as correspondent tiles.

MSER matching occurs in two steps. First, the thousands of MSERs in the finer granularity set of decomposed images tiles are successively compared to the thousands of MSERs in the coarser granularity set (Figure 5). The MSERs in each tile $X_1$ at $T_1$ is compared with each set of MSERs in a tile of $X_2$ at $T_2$. A likelihood of matching is stored for each MSER comparison. The four highest likely candidates from $X_2$ are identified (e.g., (B), (C), (D), and (E) in Figure 5). Any unmatched MSERs are preliminarily identified as potential changed regions.

3.3 SIFT Change Detection Algorithm
MSER matching generates candidates that may contain many “noisy” regions that do not represent actual changed regions. Relative to one MSER in an image with coarse granularity, a finer granularity image may generate several MSERs at the same geo-
referenced location. These MSERs are initially marked as changed regions because we need to cluster the fine-granularity MSERs for matching. Therefore we use SIFT inside the changed MSERs to refine LUCC detection. This process is composed of three steps: SIFT extraction, SIFT matching, and spatial regression voting.

SIFT extraction and feature matching are implemented using the CSIFT (Abdel-Hakim and Farag, 2006) and RANSAC algorithm, respectively. First, we identify the CSIFT key points and generate the CSIFT descriptors. Then we calculate the vector distance between the CSIFT descriptors in $X_1$ and $X_2$, to find the initial CSIFT matching pairs. Then we employ RANSAC to remove CSIFT pairs with large vector distances until we reach a predefined stop condition (usually a number of iterations or a percentage of the initial data). Finally, we obtain a number of SIFT pairs that represent similar points in $X_1$ and $X_2$.

A spatial regression voting algorithm determines whether changed MSERs represent actual LUCC. This algorithm is inspired by a SIFT voting method proposed by Zamir and Shah (2010). For the $n$ changed MSERs $\{M_1, M_2, ..., M_n\}$, the center of gravity for each MSERs, $\{g_1, g, ..., g_n\}$, is calculated. We then separately compute the vector distances from the center of gravity $g_i$ to $p$ unchanged SIFT key points and $q$ changed points inside the changed MSER $M_i$. The value of each SIFT key points $S(i)$ is defined by:

$$S(i) = \begin{cases} 1, & \text{if } S(i) \in \{\text{matched SIFT}\} \\ -1, & \text{if } S(i) \in \{\text{unmatched SIFT}\} \end{cases}$$

(2)

Voting in MSER $M_j$ is expressed as:

$$V(j) = \sum_{i=1}^{p+q} \left[ \frac{D'(i, j)}{D(i, j)} \ast S(i) \right]$$

(3)
where $D'(j)$ stands for the largest distance from the centre of gravity to the edge of MSER $j$. For each $M_i$, the value of $D'(j)$ is constant. $D(i,j)$ represents the individual Euclidean distance from each SIFT point, $i$, to the center of gravity, $g_j$. The value of $V(j)$ determines if a changed MSER represents an actual changed region ($V(j)<0$) or a false changed region ($V(j)>0$).

The more changed SIFT points appear at the center of a MSER, the more likely a MSER is considered to represent actual LUCC. The center area of MSER tends to be more stable over different levels of thresholding than the edge areas (Forssé and Lowe, 2007). Accordingly, MSER and SIFT are combined for matching to generate change maps that contain the changed/unchanged regions. These change maps may contain jagged boundaries, discontinuous edges, isolated changed pixels, and “holes” in the middle of changed areas, which will need to be addressed.

3.4 Change Map Smoothing

We employ change map smoothing to remove noisy change information, based on the assumption that LUCC is more likely to occur in connected regions rather than at disjoint small regions (Ramankutty and Foley, 1999). Change map smoothing also serves to merge the many MSERs we over-generated. For example, we may have numerous tiny grass regions inside one large forest region. Change map smoothing will merge these grass regions into a forest, because the forest occupies the majority of that area.

Change map smoothing is performed here using a Markov Random Field (MRF) grouping-smoothing process (Radke et al., 2005) as follows. According to the Hammersley-Clifford theorem (Frank and Strauss, 1986), the joint probability distribution of any MRF can be written as a Gibbs distribution:
\[
P(x) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c)
\]  
where \(x\) refers to the particular configuration of the values (intensities) of pixels in the image \(X_i\) \(\{i=1,2,...,n\}\) (we have \(n=2\) for each image pair comparison) and \(Z\) stands for the normalizing constant. \(C\) represents all the cliques in the given image. One clique \(c\) is a group of pixels whose members are mutual neighbours, and \(\phi_c(x_c)\) is called the clique potential function, which helps define the energy function to be optimized.

The image can be viewed as a combination of a “true” ground image \(X_i(x)\) and noise \(W_i(x)\):

\[
Y_i(x) = X_i(x) + W_i(x)
\]

The noise removal problem can be formulated as the minimization of \(W_i(x)\), or \(\|Y_i(x) - X_i(x)\|_2^2\) using the second norm. It is widely accepted that \(W_i(x)\) follows the Gaussian distribution, so the clique potential function is

\[
\phi_c(x) = V(x_i) = \exp \left[ -\sum_{l=1}^{m} \frac{\|y_i - x_l\|_2^2}{2\delta^2} \right]
\]

\(\delta\) stands for the deviation of \(W_i(x)\), and \(m\) is the number of pixels. The clique function \(V\) is presented as

\[
V(x_i, x_j) = \gamma \min (\|x_i - x_j\|_2^2, \beta)
\]

to model the clique neighbourhood, which penalizes the difference between adjacent nodes with threshold \(\beta\) and the weight \(\gamma\). The total number of possible change map for \(X_i\) is \(K=2^m\). We define \(H_k(x)=1\) to represent change at location \(x\) in the \(k^{th}\) change map (\(k \in K\)), while \(H_k(x)=0\) means no-change at the same location. Given \(H_k\), the change map \(X_i\) is encoded as \(X^1_i\) and \(X^0_i\), which represent the change and no-change areas in \(X_i\) respectively. The conditional MRF model becomes
\[ P(x_i, x_j | H_k) = \frac{1}{Z} \left\{ \exp \left[ -\sum_{C \in E} V(x_i) - \sum_{C \in E} V(x_j) \right] \right\} \]

The associated optimization problem as shown in (9) results in an optimized change map, with the energy function in (10) obtained by merging (6) and (7) into (8).

\[ H_k = \arg \{ \max_{H_l} [\sum_{x_i, x_j \in E} P(y_i | x_i)P(y_j | x_j)P(x_i, x_j | H_l)P(H_l)] \} \]  

\[ E = -\log \left\{ \sum_{x_i, x_j \in E} \exp \left[ -\frac{1}{2\sigma^2} (y_i - x_i)^T(y_i - x_i) \right] \right\} \]

Since simulated annealing optimization follows naturally from this MRF model (Kasetkasem and Varshney, 2002), it was used to generate the optimized changed region boundaries. We begin with the original change map. We estimate its initial parameters and set the initial temperature for the simulated annealing. We then obtain a new change map from the previous change map, based on a Gibbs sampling procedure (Gerhard, 1995). Finally, we reduce the “temperature” with a predetermined schedule and repeat the prior step until there is a convergence or the maximal number of iterations is reached.

Here the temperature is the control parameter of the randomness generator for change area boundaries. More details about this algorithm can be found in the pseudo code in Appendix III. Figure 6 indicates that MRF-based change map smoothing removes small vehicles, trees, and shadows. Some large vehicles and shadows still exist after the smoothing. Large shadow areas are difficult to verify without further reference datasets since shadows are very similar to the pavement within RGB color space. It is possible to add other smoothing algorithms to remove large vehicles and shadows but those run the
risk of eliminating actual LUCC. Removing shadows and vehicles with minimum impact on LUCC in dense urban area remains an ongoing challenge (Yin et al., 2015).

Figure 6. Change map smoothing. (A) The change map generated by MSER and SIFT matching, by comparing 2005 0.11m MMCO and 2007 0.13m MMCO collected at downtown Montreal, and overlaid with the MMCO image tile in 2007; (B) The change map after the MRF-based map smoothing process. We note some large vehicles and shadows still exist after smoothing.

3.5 LUCC Labelling
Labelling LUCC is also challenging as it requires coordination between spatial and temporal scales. Labelling require significant training data and continuous landscape monitoring (Chen et al., 2012). In the following empirical study, only RS data within the Greater Montreal Area from 2005-2012 was collected and the images were only recorded in early July to avoid seasonal differences. Consistent with practice, standard land use or land cover labels are used (Ridd and Liu, 1998). For image classification, a Support Vector Machine (SVM) classifier is selected due to its high accuracy and low sensitivity to noisy data in RS research (Melgani and Bruzzone, 2004). Seven labels for the SVM classification are used: forest, grass, farmland, bare ground, water, roads and buildings. A subset of raw images is used for classifier training and then applied to the rest of the
imagery datasets. Finally, ground truth data is employed to evaluate the accuracy of the scale-invariant LUCC detection method.

4. Case Study in Montreal LUCC
The scale-invariant LUCC detection method was evaluated in the Greater Montreal Area, Quebec, Canada. The area covers approximately 4,163 km² from 2005-2012. Details about the scale heterogeneous data were listed in Table 1. The 2005 Montreal Metropolitan Community Orthophotos (MMCO) covered most urban areas in the City of Montreal. To obtain a seamless image for 2007, we used MMCO for the most areas of Montreal city and surroundings at 0.13m spatial granularity; some suburban and rural areas of the Greater Montreal Area were recorded at 0.3m spatial granularity. Computing resource provisioning was supplied by a hybrid cloud composed of one local controller (Intel® Core™ i7-6700 Processor, 32GB memory, and 2TB storage) and Virtual Machines (VMs) from the Microsoft Azure cloud computing platform. Four Azure Hadoop clusters were utilized for four cross-scale LUCC processes, with each cluster consisting of six VMs. Most of the code was developed in Java, based on Hadoop, BoofCV, and OpenIMAJ libraries. The detailed implementation of the workflow was illustrated in Figure 7.

The scale-invariant LUCC detection method was designed for image pair comparisons so there were four separate LUCC comparisons 2005-2006, 2006-2007, 2007-2009, and 2009-2012. Azure D3_V2 VM was chosen for the LUCC 2005-2006 process (4 cores and 14GB memory). Both 2006-2007 and 2007-2009 processes utilized six D5_V2 VMs (16 cores and 56GB memory). The 2009-2012 comparison was deployed on six D4_V2 nodes (8 cores and 28GB memory). The VM configurations were determined by the trade-off between the computing workload and costs (Zhu and
Agrawal, 2010). Five hundred GB Azure online file storage (100GB for each year; 0.13m data was selected for most areas, and 0.3m data for the other areas, for comparison of the two 2007 datasets with different granularities) was utilized for the datasets in Table 1\(^1\).

All the raw datasets were geo-referenced; consistent with computer vision, the tiles were not.

For each LUCC comparison, the scale-invariant LUCC detection workflow was deployed as five MapReduce steps. The first map step extracted the MSERs; whereas the MSER matching was conducted as a reduce step (Section 3.2). The second map step extracted SIFT; whereas the reduce computation removed “artificial” SIFT features (e.g., artificial SIFT features can be caused by tile borders, as the artificial border challenge in \{Xing et al., 2014\}). The third map step deployed the SIFT matching and the spatial regression voting algorithm (see Appendix II for the pseudo code for the spatial regression voting). The fourth map step handled change map smoothing (Section 3.4). There was no reduce steps for the third and fourth MapReduce process. The final map step scheduled the SVM classification (Section 3.5) and recombined the distributed results and output the final results to the local controller in its reduce step.

\(^1\) Although the scale-invariant LUCC method is designed to solve the big data challenges in LUCC, the data in our case study is not so big (~500GB), due to the limited availability of high-resolution RS imagery data at the Schulich Library of McGill University.
**Figure 7.** Implementation of the scale-invariant LUCC workflow for our Montreal urban-rural LUCC case study.

**Table 1.** Details about datasets used in the Montreal urban-rural LUCC

<table>
<thead>
<tr>
<th>Year</th>
<th>Platform</th>
<th>Spatial Resolution (m)</th>
<th>Spectral Resolution</th>
<th>Spatial Extent (km²)</th>
<th>Number of Image Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>Montreal Metropolitan Community Orthophotos</td>
<td>0.11</td>
<td>RGB (sharpened &amp; fused)</td>
<td>75.02</td>
<td>62</td>
</tr>
<tr>
<td>2006</td>
<td>DMTI</td>
<td>0.60</td>
<td>RGB (sharpened &amp; fused)</td>
<td>3528.00</td>
<td>50</td>
</tr>
<tr>
<td>2007</td>
<td>Montreal Metropolitan Community Orthophotos</td>
<td>0.13 / 0.30</td>
<td>RGB (sharpened &amp; fused)</td>
<td>3718.75 / 139.73</td>
<td>2380/18</td>
</tr>
<tr>
<td>2009</td>
<td>DMTI</td>
<td>0.60</td>
<td>RGB (sharpened &amp; fused)</td>
<td>2257.92</td>
<td>32</td>
</tr>
<tr>
<td>2012</td>
<td>DMTI</td>
<td>0.60</td>
<td>RGB (sharpened &amp; fused)</td>
<td>4163.04</td>
<td>59</td>
</tr>
</tbody>
</table>

The entropy-based spatial decomposition was illustrated using MMCO 2005 and DMTI 2006 datasets. Each MMCO image file covered approximately 1.21 km² area; whereas the DMTI image covered nearly 70.56 km². The average of the first part of $E$, $(H = -\sum_{i=1}^{n} P_i \log_2 P_i)$ of the entropy in (1) was 7.51 and 6.74 for MMCO 2005 and DMTI 2006, respectively. Then $|E_1 - E_2| < \tau$ became $|7.51 \ast \log_2 extent^1 - 6.74 \ast$
\[ \log_2 \text{extent}^2 + 18.64 \] < 10. We chose \( \tau = 10 \). If \( H \) was the same for any image pair then the pixel difference between image tiles would be no more than 1024 \( (2^{10}) \), which appeared to provide a satisfactory tolerance for creating tile pairs that generated similar number of SIFT points and MSERs. We obtained 0.24 km\(^2\) and 1.88km\(^2\), as the extent of the decomposed image tiles, respectively. The spatial decomposition was depicted in Figure 8. Since the number of tiles must be an integer, 2*2 and 6*6 decomposition were selected in Figure 8 as the closest solution.

![Spatial Decomposition](image)

**Figure 8.** The spatial entropy-based spatial decomposition. (A) 2*2 splitting of MMCO imagery data; and (B) 6*6 decomposition of DMTI dataset.

To generate a larger number of smaller MSERs, the MinArea was set to 10 and MaxVariation equaled 0.2 for the MSER extraction in Section 3.2. Following Forssén (2007), the \( m_{\text{min}} \) parameter was set to 0.003 and the step parameter \( n \) was heuristically set to 200 (i.e., we preferred more iterations with a smaller threshold difference that would generate more but smaller MSERs). We favored generating a large number of small MSERs as opposed to a smaller number but bigger MSERs. This reduced the risk of missing LUCC, but can result in more noise. After MSERs were extracted from the decomposed tiles, we implemented MSER matching for the potential changed MSER identification. The MSER extraction and matching were implemented with OpenIMAJ.
library (Hare, Samangooei, and Dupplaw, 2011). The pseudo code for MSER matching was listed in Appendix I.

Tile comparison was a many-to-many process, which would create a problem in serial processing but MapReduce implementation turned it into a parallel one-to-many matching, based on the \(<key, value>\) data structure. Separate change maps were favored over fused change maps that can depict LUCC at a finer granularity but conceal the scale variance. Consequently there were two versions of change maps for the year 2006, 2007, and 2009, generated from the different LUCC comparisons.

For both MSER and SIFT matching, the vector distance ratio method was adopted for initial matching, with 1.5 as the threshold. RANSAC was then implemented by fitting a geometric model— an affine transform model was chosen —to the initial results (Pereira and Pun, 2000) since the matches of image features could be invariant to translations, rotations and scale changes. This process iteratively selected a random set of matches, estimated the geometric model from the selected random set and then tested the remaining matches against the learned model – always eliminating outliers. The method looped until the size of matches reached below 50 percent of the initial match. The RANSAC matching was based on the OpenIMAJ library. The change map smoothing algorithm in Section 3.4 removed noisy change information (see Appendix III for the pseudo code). The parameters in the above steps were determined using the sampling strategy from the four LUCC processes. In each comparison, five image pairs (without decomposition) were sampled to calculate the parameters. The classification was developed on libSVM (Chang and Lin, 2011) with seven predefined labels in Section 3.5. The SVM training process was conducted according to Melgani and Bruzzone (2004).
with image features of MSERs, using brightness, shape, and texture. The SVM classifier training processes employed the same five image pairs.

The results of the LUCC comparisons were verified with 1000 ground truth points. We collected 650 points with purposive sampling in high density areas (i.e., downtown and rapidly developing areas like the City of Laval) because we wanted to verify our method in problem areas (e.g., tall buildings with long shadows and road repaving). The other 350 points were collected via random sampling by gridding the Great Montreal Area. All 1000 points were physically inspected. We note the oversampling of problems likely negatively impacted the accuracy compared with a completely random sampling. The accuracy of the LUCC was shown in Table 2, which was obtained by geo-registering the LUCC results with the ground-truthing data and We examined any regions intersecting with the ground-truthed points. If none intersected then we selected the region with the nearest Euclidean distance.
Table 2. LUCC Accuracy Evaluation with 1000 Ground-Truthing

<table>
<thead>
<tr>
<th>Ground-Truthing</th>
<th>Change (%)</th>
<th>NoChange (%)</th>
<th>Total (%)</th>
<th>User's Accuracy (%)</th>
<th>Total Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005-2006 LUCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>6.6</td>
<td>36.1</td>
<td>42.7</td>
<td>15.5</td>
<td>62.4</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>2.1</td>
<td>55.2</td>
<td>57.3</td>
<td>96.3</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>8.7</td>
<td>91.3</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>75.9</td>
<td>60.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007-2006 LUCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>4.3</td>
<td>24.6</td>
<td>28.9</td>
<td>14.9</td>
<td>67.9</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>6.6</td>
<td>64.6</td>
<td>71.1</td>
<td>90.9</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>10.9</td>
<td>89.2</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>39.4</td>
<td>72.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007-2009 LUCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>12.1</td>
<td>21.3</td>
<td>33.4</td>
<td>36.2</td>
<td>72.9</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>5.8</td>
<td>60.7</td>
<td>66.5</td>
<td>91.2</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>17.9</td>
<td>82.0</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>67.6</td>
<td>74.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2009-2012 LUCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>3.0</td>
<td>5.7</td>
<td>8.7</td>
<td>34.5</td>
<td>85.1</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>9.2</td>
<td>82.1</td>
<td>91.2</td>
<td>90.0</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>12.3</td>
<td>87.7</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>24.3</td>
<td>93.6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3. LUCC Accuracy Evaluation with 350 Random Ground-Truthing

<table>
<thead>
<tr>
<th>Ground-Truthing</th>
<th>Change (%)</th>
<th>NoChange (%)</th>
<th>Total (%)</th>
<th>User's Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005-2006 LUCC</td>
<td>8.3</td>
<td>28.6</td>
<td>36.9</td>
<td>22.5</td>
</tr>
<tr>
<td></td>
<td>3.7</td>
<td>59.4</td>
<td>63.1</td>
<td>94.1</td>
</tr>
<tr>
<td></td>
<td>12.0</td>
<td>88.0</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>69.1</td>
<td>67.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007-2009 LUCC</td>
<td>5.7</td>
<td>21.3</td>
<td>27.0</td>
<td>21.1</td>
</tr>
<tr>
<td></td>
<td>3.1</td>
<td>70.0</td>
<td>73.1</td>
<td>95.6</td>
</tr>
<tr>
<td></td>
<td>8.8</td>
<td>91.3</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>64.8</td>
<td>76.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2009-2012 LUCC</td>
<td>14.9</td>
<td>20.2</td>
<td>35.1</td>
<td>42.6</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>63.7</td>
<td>64.8</td>
<td>98.3</td>
</tr>
<tr>
<td></td>
<td>16.0</td>
<td>83.9</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>93.1</td>
<td>75.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.9</td>
<td>9.4</td>
<td>12.3</td>
<td>23.6</td>
</tr>
<tr>
<td></td>
<td>10.6</td>
<td>77.1</td>
<td>87.7</td>
<td>87.9</td>
</tr>
<tr>
<td></td>
<td>12.5</td>
<td>87.5</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td>23.3</td>
<td>88.1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4. LUCC Accuracy Evaluation with 650 Purposive Ground-Truthing

<table>
<thead>
<tr>
<th>Ground-Truthing</th>
<th>Change (%)</th>
<th>NoChange (%)</th>
<th>Total (%)</th>
<th>User's Accuracy (%)</th>
<th>Total Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2005-2006 LUCC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>5.7</td>
<td>40.2</td>
<td>45.9</td>
<td>12.4</td>
<td>58.4</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>1.2</td>
<td>52.9</td>
<td>54.1</td>
<td>97.8</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>6.9</td>
<td>93.1</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>82.6</td>
<td>56.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2007-2006 LUCC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>3.5</td>
<td>26.3</td>
<td>29.8</td>
<td>11.7</td>
<td>65.2</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>8.5</td>
<td>61.7</td>
<td>70.2</td>
<td>87.9</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>12.0</td>
<td>88.0</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29.2</td>
<td>70.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2007-2009 LUCC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>10.6</td>
<td>21.8</td>
<td>32.4</td>
<td>32.7</td>
<td>69.7</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>8.5</td>
<td>59.1</td>
<td>67.6</td>
<td>87.4</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>19.1</td>
<td>80.9</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>55.5</td>
<td>73.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2009-2012 LUCC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change (%)</td>
<td>3.1</td>
<td>3.7</td>
<td>6.7</td>
<td>46.3</td>
<td>87.9</td>
</tr>
<tr>
<td>NoChange (%)</td>
<td>8.5</td>
<td>84.8</td>
<td>93.3</td>
<td>90.9</td>
<td></td>
</tr>
<tr>
<td>Total (%)</td>
<td>11.6</td>
<td>88.5</td>
<td>100.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Producer's Accuracy (%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>26.7</td>
<td>95.8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In the three tables of accuracy assessment, our lowest total accuracy occurred in the 2005-2006 LUCC comparison. Most error derived from false change areas (36.1% in Table 2). Numerous regions in the MMCO 2005 were designated as changed since they
failed to match regions in DMTI 2006. Some error came from subtle view angle differences, which caused some building windows to be visible only in images at higher resolutions. These generated several false changes. Error might also come from noisy information (e.g., vehicles, trees, and water on the roads), which was frequently observed in dense urban areas. Table 3 displayed higher total accuracy than Table 4 in the 2005-2006 comparison, because our purposive sampling was conducted in the dense urban areas.

The highest total accuracy in the three tables was achieved from the 2009-2012 comparison because the two datasets were recorded with the same spatial resolution and sensing platform. For the 2007-2006 and 2007-2009 LUCC comparisons, the total accuracy was 68.9 percent and 72.8 percent, as shown in Table 2. The small difference between the accuracies can be explained by the larger spatial extents covered by the DMTI 2006 dataset. The reason for accuracy differences, we believe, not only lied in the scale variance (i.e., spatial granularities and extents) of data, but also in smaller differences in view angle, shadow, vehicle, trees, and water areas of MMCO and DMTI data. The coarser spatial resolution of DMTI data reduced shadow and vehicle noise to some extent. It was important to remember that high resolution imagery datasets did not guarantee high accuracy in LUCC as high resolution inevitably generates more diverse and noisier information.

We also delineated the accuracy assessment with random and purposive ground-truthing in Table 3 and Table 4, respectively. The average of total accuracy in random sampling was higher than the results of purposive sampling (except 2009-2012), which further confirms the difficulty of studying LUCC in dense urban areas. These areas
contain a high number of building, roads, forest, grass, and noisy information (e.g., shadows and vehicles). We also note the purposive ground-truthing was helpful understanding urban LUCC (Grădinaru, Kienast, and Psomas, 2017). It allowed us separately assess areas like buildings and roads instead of relying on the random sampling approach (e.g., only forest changes).

We compared our total accuracy to other LUCC studies. An MSER and SURF-based LUCC detection achieved approximately 75 percent total accuracy (Ye et al., 2014). They utilized resampling to hide the scale heterogeneity of 200 aerial photos. Their accuracy resembles our 2007-2009 comparison; however our method did not require preprocessing the data with resampling. Raja et al. (2013) achieved 82.5 percent accuracy in their scale-variant LUCC study (IRS-1B at 72.5m compared to IRS-P6 at 5.8m) but they also employed resampling. Pham, Mercier, and Michel (2016) reported 85 percent total accuracy, using a SIFT matching and graph-based LUCC detection method. They conducted their test using a pair of 800 × 400-pixel radar images, each at the same (10m) resolution. Their result was similar to our 2009-2012 comparison. This suggests that our scale-invariant LUCC detection method can handle the scale heterogeneity directly and still achieve good results. To further improve the accuracy, possible solutions could entail resampling the results to the same granularities, or utilizing image fusion techniques (Li, Manjunath, and Mitra, 1995) to homogenize the scale of the results.

Although portions of our empirical study did not generate very high accuracy, we argue that the scale-invariant LUCC detection method can more effectively extract LUCC from scale heterogeneous datasets without image scaling techniques (e.g., spatial interpolation, down-sampling, and image resizing). Image scaling techniques invariably
assign pixel values that are consistent with their neighbours, which increases the risk of missing LUCC (Dai and Khorram, 1998). Large scale differences (both granularity and extent) will likely lower LUCC detection accuracy. The scale-invariant LUCC detection method can reduce the influence of scale variance in LUCC detection but not eliminate it.

5. Conclusion
This paper presents the promises and challenges of handling scale variance in LUCC and proposes a scale-invariant LUCC detection method. Our method integrates extent, shape, and spectral information into scale-invariant image features with a workflow composed of entropy decomposition, MSER, SIFT, spatial regression voting, change map smoothing and LUCC labelling. The method is deployed in a cloud computing and Hadoop framework to address the scale variance challenges in big data. The case study used five scale heterogeneous imagery datasets in Greater Montreal, from 2005 to 2012, to demonstrate the potential for our scale-invariant LUCC detection method. Although the scale difference of our datasets varied by greater than five times (0.6m in 2006 versus 0.11m in 2005), our method achieved reasonable LUCC accuracy. Overall, the accuracy of our method reached similar levels as other studies, without resorting to resampling.

We note drawbacks of our method. First, not all changed regions can be extracted as MSERs (e.g., construction sites and the road repair works) when these regions cover small spatial extents (for us, less than minArea) and are unstable across different levels of intensity thresholds. Second, some features (e.g., road re-pavement with darker colors in the image) are difficult to distinguish from shadows, due to similarities in shape and spectral attributes. Third, noisy objects with well-defined borders and sharp contrasts from their neighbouring objects (e.g., large vehicles) produce unmatched MSERs with unmatched SIFT points. As these are not LUCC, the total accuracy is decreased. These
drawbacks increase as the image granularity difference increases (Haverkamp and Poulsen, 2003).

Big data has significantly changed LUCC research, not only in terms of data management and processing, but also on spatial-temporal scales. Short time period changes can be captured by advances in sensing platforms (e.g., the temporary construction sites). We assume that modelling of both spatial and temporal variance in LUCC will focus increasingly on temporal analysis. Future research will investigate methods that integrate spatial and temporal variance to build consistent spatial-temporal LUCC models.

Acknowledgement
This research has been funded by a Microsoft Azure Research Award and the Social Sciences and Humanities Research Council, Canada (SSHRC).

References


Haghani, A., Hamedi, M., Sadabadi, K., Young, S., & Tarnoff, P. (2010). Data collection of freeway travel time ground truth with bluetooth sensors. Transportation Research Record: Journal of the Transportation Research Board, (2160), 60-68.


Appendix I: MSER Matching in Recomposition

Algorithm: MSER Matching

**Input:** the image lists \( A \) and \( B \), the MSER lists \( MA \) and \( MB \), and \textit{threshold} for MSER matching

**Output:** the list \( L \) containing the correspondence between \( A \) and \( B \)

for each MSER \( ma \) in \( MA \):

\[ S = \text{zeros(size}(B)) \]

for each MSER \( mb \) in \( MB \):

\[ s_i = \text{match}(ma.score, mb.score) \]

\( S.add(s_i) \)

end for

\( S' = \text{descend_sort}(S) \)

\( S' = \text{sub_list}(S', 1, 4) \)

for \( i = 1:4 \)

if \( s_i \leq \text{threshold} \)

\( S'.remove(i) \)

end if

end for

\( L.add(S') \)

end for

return \( L \)

End
Appendix II: SIFT Change Detection within Voting

Algorithm: SIFT Change Detection with Voting

**Input:** image tile $I$, unmatched MSER list $UM$ for $I$, and SIFT list $S$ for $I$

**Output:** the MSER change region list $C$

for each unmatched MSER $u$ in $UM$:

\[
  u.\text{gravity\_center} = \left(\frac{\max(u.x) + \min(u.x)}{2}, \frac{\max(u.y) + \min(u.y)}{2}\right)
\]

\[
  u.\text{score} = \text{Equation (5.3) in Section 5.3.3 using } S \text{ and } u.\text{gravity\_center}
\]

if ($u.\text{score} \leq 0$)

$C.\text{add}(u)$;

end if

end for

return $C$

**End**
Appendix III: Change Map Smoothing

Algorithm: Change Map Smoothing

**Input:** initial image change map $C$, and the original imagery dataset $D$.

**Output:** smoothed image change map $C$

for $i = 1$ to Max_Iteration

$T = T_0 \log(1 + i)$

for $k = 1$ to Max_k

for $m = 1$ to Max_m

if ($C(k,m) = 0$)

$E_0 = \text{Equation}(10) \text{ in Section 3.4}$

else

$E_I = \text{Equation}(10) \text{ in Section 3.4}$

end if

$P_0 = \exp(-E_0/T)$

$P_I = \exp(-E_I/T)$

$P_0 = P_0 \times (P_0 + P_I)$

$R = \text{rand}(0,1)$

if ($R < P_0$)

$C(k,m) = 0$

else

$C(k,m) = 1$

end if

end for

end for

end for

return $C$

End