Point cloud watermarking

doc. dr. Bogdan Lipuš

Laboratory for Geospatial Modelling,
Multimedia and Artifical intelligence

Wednesday 9th June, 2021

Watermarking - Introduction

- Two task:
  - **Embedding** (inserting) some information (watermark) into the data;
  - **Detection/Extraction** of this information from the data;
  - (secret) **key** determines locations of this embedding information;
- Primarily, the watermarking is developed for LiDAR data (LAS files):
  - geo-referenced points; most important are the the coordinates of the points which represents the real terrains; other attributes.
- Extended to the 3D point clouds:
  - positioned, oriented and scaled in many ways;

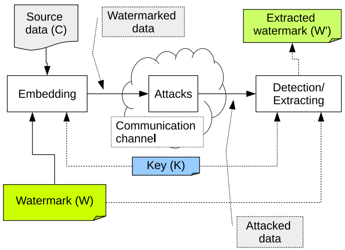


Figure 1: Watermarking process

Watermarking - Introduction

- The coordinates of the point were choosed for watermarking the point clouds:
  - They are slightly modified (unnoticeable) but they need to be still resistible against various other data modifications (attacks);
- The data (3D point cloud) can be modified in various ways:
  - Destroy information (watermark), embedded into the data $\Rightarrow$ thus, the data are attacked;
- The following attacks can occur:
  - Affine transformations (scaling, rotating, translating), simplification (voxelization), noise, random point removal, cropping, points rearrangement etc.;
  - For geo-referenced LiDAR data, the affine transformations were not considered;

- The watermark **embedding** procedure:
  1. Defining an array of marker positions (where the particular watermark bit would be embedded),
  2. Inserting a watermark bit within the circular area around the marker positions, and
  3. Modifying the coordinates of the points within this circular area.
- The watermark **extraction** procedure is very similar:
  1. Defining an array of marker positions (where particular watermark bit was embedded), and
  2. Extracting a watermark bit from these circular areas around marker positions.

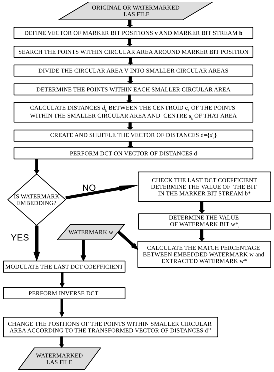


Figure 2: Embedding/Extraction diagram

Watermark embedding - markers

- A watermark w:
  - $w = \{w_0, w_1, \ldots, w_{M-1}\}, \quad w_i \in \{0, 1\}$
- A watermark bit stream b:
  - $b = \{b_0, b_1, \ldots, b_j, \ldots, b_{N-1}, \}$
  - where: $b_j = w_{(j \bmod M)}$
  - thus, the watermark is embedded multiple times ($N = MX$);
- For each b_i: a marker positioned v_i is randomly defined and the corresponding circular area (within radius r_{out});
- Next, the smaller circular areas (seeds s_k) are determined applying the sunflower seed algorithm;

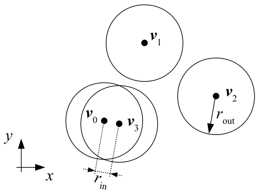


Figure 3: Marker bit positions

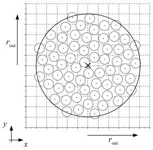


Figure 4: The sunflower seed algorithm

1. Determine all points within the smaller circular area (seed) by the approximate nearest neighbor search (FLANN);
2. Determine the geometric center $\mathbf{c}_k$ of all these points, and calculate distance d_k between $\mathbf{s}_k$ and $\mathbf{c}_k$;
3. The distances d_k for each marker position $\mathbf{v}_k$ was obtained; further shuffled by the Mersenne twister pseudo-random generator;
4. d_k are used as input for the DCT (Discrete cosine transformation);
5. Modify the last DCT coeficient (0 or 1);
6. Perform IDCT, thus, modified d'_k are obtained $\Rightarrow c_k$ and the points within the smaller circular area are changed;

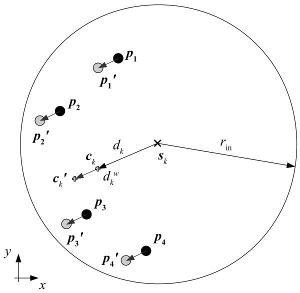


Figure 5: Modifying the points' coordinates within small circular area

1-4. The steps identical to the steps in the watermark embedding $\Rightarrow$ the distances d_k for each marker position are determined, and the DCT is performed;
5. The last DCT coefficient is checked (the bit value 0 or 1 is determined);
6. Two counters cw_i^+ and cw_i^- are defined for each watermark bit w_i;
    - they counts how many times occur 1 or 0 for w_i within the watermark bit stream b;
    - if $cw_i^+ > cw_i^- \Rightarrow w_i = 1$ otherwise $w_i = 0$;

- In general, 3D point cloud can be positioned, scaled, and oriented in different ways;
- Thus, before extraction the point clouds should be aligned (registered);
- Unfortunately, the most point cloud registration techniques works only for the point clouds with the same scale;

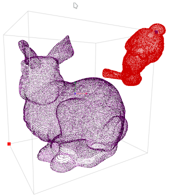


Figure 6: Point cloud registration issue

Convex hull point cloud registration

1. The two convex hulls H^W (watermarked and possibly attacked cloud) and H^I (original cloud) $\Rightarrow$ two arrays of triangles T_w and T_i are obtained and sorted according to the triangle area in descending order;
2. For each triangle from T_w, the matching triangles from the T_I are determined (can be more than one):
   - the ratios between the triangle sides are used (a/c and b/c);
3. Next, the best matching triangles are determined:

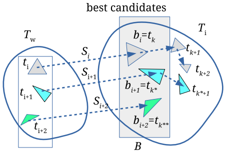


Figure 7: Matching triangles

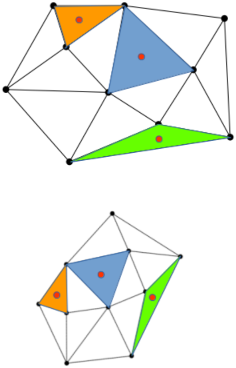


Figure 8: 3D Convex hulls

Convex hull point cloud registration

3. The best matching triangles are determined (cont.):
   - In general, the largest triangles from T^W with the similar ratios match with the largest triangles from T^I;
   - ☹Any modifications of the watermarked point cloud can damage the convex hull;
4. The scaling factor s is determined applying the following steps:
   1. The scaling factors (b_i) are sorted in the ascending order;
   2. The scaling factor b_i are joined together within $[-\xi, \xi]$, and the scaling average s_i and the number of them is determined as the frequency f_i;
   3. The final step: the scale factors s_i with high frequencies f_i that are close enough $\Rightarrow$ they are joined together into the final scale $s = s_k$ with the highest frequency s_k;

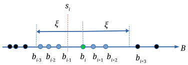


Figure 9: Array of scaling factors S_i binds the best candidates $(b_i, b_{i+1}$, and $b_{i+2})$ for scale estimation

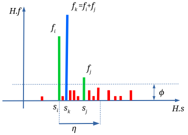
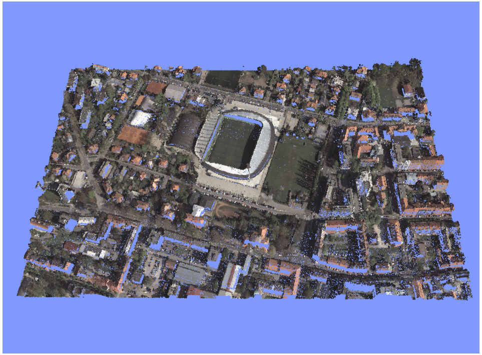


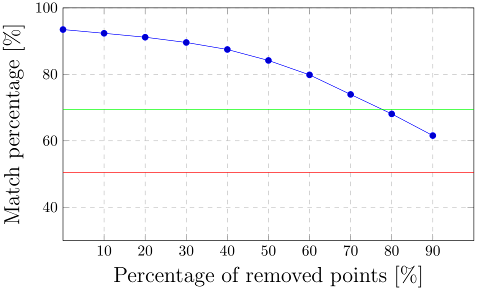
Figure 10: Determining scale factor s_i and f_i; s_i in an average of $f_i = 6$ scale factors in this example

Convex hull point cloud registration

5. Two auxiliary point clouds are built from the centres of best matching triangles of the convex-hulls, respectively (marked as red in Figure 8); much less points;
6. The fisrst auxiliary point clouds (from watermarked point cloud) is scaled by s and the rigid transformation matrix (i.e rotation and translation) is determined and applied to the watermarked cloud;
7. This coarse (initially) aligned point cloud is then aligned by well known algorithm Iterative closest point (ICP).



(a)



(b)

Figure 11: (a) After random removal attack (60% of random removed points; match percentage m^p : 87.50%)
(b) Random removal attack

(a) (b)

Figure 12: (**a**) Affine transformation ($m^p = 100\%$). (**b**) Affine transformation and cropping ($m^p = 95.31\%$).

Figure 13: (**a**) Affine transformation, cropping and random removal ($m^p = 93.75\%$). (**b**) Affine transformation and local noise ($m^p = 96.88\%$).

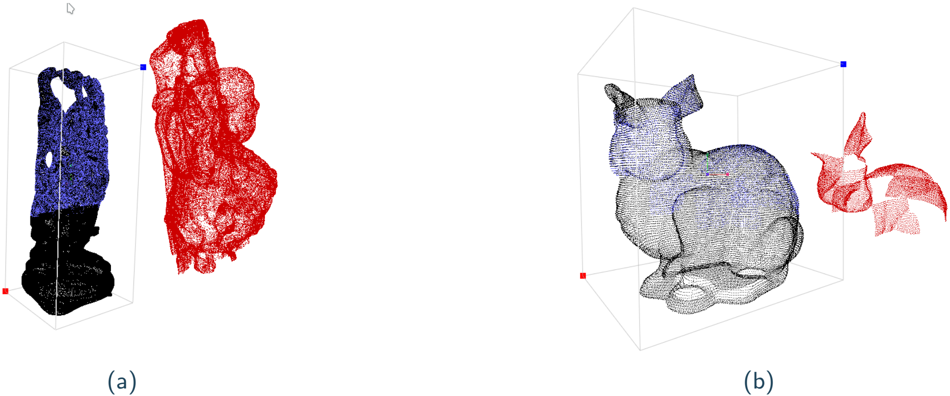


Figure 14: (**a**) Affine transformation and cropping with slantwise cut ($m^p = 87.50\%$). (**b**) Affine transformation and complex cropping ($m^p = 93.51\%$).

More can be read:

Bogdan Lipuš and Borut Žalik.
Robust watermarking of airborne LiDAR data.
Multimedia Tools and Applications, 77(21):29077–29097, 2018.

Bogdan Lipuš and Borut Žalik.
3D convex hull-based registration method for point cloud watermark extraction.
Sensors, 19(15):3268, 2019.

Questions

???