Reconstruction of relative 3D positioning of horse's bones

Summer Internship

Jilliam María DÍAZ BARROS

2013

Advised by:

A.Mansouri

C.DEMONCEAUX

A.Habed

Université de Bourgogne

Table of Contents

Abstract

The archeologists' interest lies in studying rites performed in Iron Age societies through the analysis of spatial arrangements of excavated relics and, more importantly, animal skeletons. The spatial arrangement of animal bones may indeed shed some light on certain religious practices. This generally requires working on replicas of the excavated bones, or better, digitized instances of these. To this end, a comprehensive library of 3D models representing the entire skeleton of a horse obtained using a 3D laser scanner has been built. However, in order to recover the entire 3D model of the animal's skeleton, the digitized bones need to be positioned in space by relying on photographs captured in situ while documenting the excavation process. The goal of this project is to employ Computer Vision techniques to exploit the available images, bone models and additional bone maps as to determine the relative 3D positioning of the bones as found in the pit.

Problem statement

The primary task in the present project is to determine the spatial arrangement of bones of horses through a set of images acquired during the excavations, and obtain the 3D reconstruction of the scene. However, the images were acquired at different levels of the excavations, with different scene lighting, and some bones might be incomplete or partially occluded by other bones or sand. Furthermore, the calibration of the camera was not available and according to the EXIF tags, the focal length was modified when acquiring different views of the bones arrangement.

Proposed Approach

Considering previous limitations, a semi-supervised method for the 3D reconstruction of the scene is presented. The proposed approach can be summarized in five steps, depicted in Figure 1:

1. Object extraction.
2. Feature detection and matching.
3. Generation of additional points.
4. 3D reconstruction.
5. Point cloud matching.

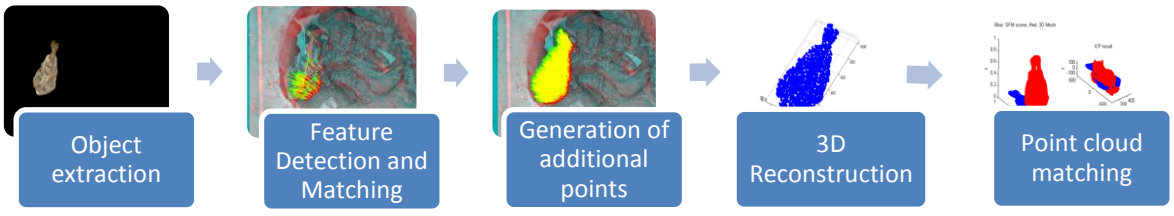

Figure 1. Proposed approach

The implemented steps are explained below.

It is important to mention that for the 3D reconstruction at least two views of the scene had to be available. Steps 1, 2, 3 and 4 require the information in those pairs of images.

Object extraction

The first step was to specify the location of each bone within each image, and subtract the background. In order to do so, a binary mask for each bone was extracted manually. These masks were created for each bone in each pair of images.

One example of this procedure can be observed in Figure 2.

a. b. c.

Figure 2. a. Original image. b. Binary mask indicating the location of the skull. c. Skull extracted using the binary mask.

Feature detection and matching

As a second step, the feature points of the resulting images were extracted using the following methods:

- Features from Accelerated Segment Test (FAST) by Rosten & Drummond [1].
- Speeded-Up Robust Features (SURF) by Bay *et al* [2].
- Maximally Stable Extremal Regions (MSER) by Matas *et al* [3].
- Harris corners by Harris & Stephens [4].
- Minimum eigenvalue by Shi & Tomasi [5].

A function '`detectFeatures`' was created to perform all the operations.

All these methods were implemented since the texture in some bones was homogeneous, and consequently the number of features points was sparse.

From these points, features vectors were extracted using SURF [2]. To perform the feature matching in each pair of images, a function '`featuresMatching`' was created. Within this function, the outliers were eliminated using RANSAC.

Generate additional points

In most of the cases, regardless of using different algorithms of point detection, the number of feature points were reduced and not uniformly distributed in the whole region. This led to some mismatching during the comparison of the clouds of points.

In order to avoid the previous problem, additional points were added uniformly to the area of interest in one image, and projected in the second one assuming an affine transformation *F* between the two images (see equation (1)) [6]. In this equation (*u,v*) and (*x,y*) are the coordinates of the same point in two images. This process was performed taking into account that the number of matched points was at least 5.

$$\begin{pmatrix} u \\ v \end{pmatrix} = F \begin{pmatrix} x \\ y \end{pmatrix} \tag{1}$$

Since it is assumed an affine transformation, equation (1) can be expressed as (2).

$$\begin{pmatrix} u \\ v \end{pmatrix} = s \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{2}$$

If $a = \cos(\theta)$ and $b = \sin(\theta)$, (2) can be represented as the linear system presented in (3), or $Ax = b$, and can be solved using Linear least squares (4).

$$\begin{pmatrix} x & -y & 1 & 0 \\ y & x & 0 & 1 \\ & \dots & & \end{pmatrix} \cdot \begin{pmatrix} a \\ b \\ t_x \\ t_y \end{pmatrix} = \begin{pmatrix} u \\ v \\ \vdots \end{pmatrix} \tag{3}$$

$$\hat{x} = (A^T A)^{-1} A^T b \tag{4}$$

Within the code, *F* was computed in the function '`findAffineTransf`'.

3D Reconstruction

Synthetic Scene

At first, a synthetic scene was created to test different reconstruction algorithms. For these tests, one of the 3D models of the provided files was used. The intrinsic and extrinsic parameters were assumed to be known. Two resulting images were computed using (5), where X represents the coordinates of the points in the 3D world and *x* the 2D points in the pixel coordinates [6].

$$x = K[R|T]X \tag{5}$$

The essential matrix was computed with the following equation:

$$E = [\mathbf{t}]_x R$$

And the fundamental matrix, using the essential matrix:

$$F = K^{T^{-1}} \cdot E \cdot K^{-1}$$

The 3D scene X_n was estimated with the Direct Linear Transformation method, using the projection matrices P and P', and the coordinates of the points in both images x, y, 1) and (x', y', 1)], with equation (6):

$$\begin{bmatrix} \mathrm{x}p^{3T} - p^{1T} \\ \mathrm{y}p^{3T} - p^{2T} \\ \mathrm{x}'p'^{3T} - p'^{1T} \\ \mathrm{y}'p'^{3T} - p'^{2T} \end{bmatrix}_{4\times4} X_{n_{4\times1}} = 0_{4\times1} \qquad (6)$$

X_n was computed using SVD of the 4x4 matrix, and selecting the singular vector of the smallest singular value [6].

Later on, the back projection was performed to compute the residual error using the projection matrices:

$$x = PX_n$$

The process was repeated one more time, computing the fundamental matrix from the feature points, using the MATLAB function `estimateFundamentalMatrix`. In this case, it was assumed that the essential matrix was not available.

For the selected mesh ('PatArDrFémur.obj'), the error using the fundamental matrix obtained from the essential matrix was of 0.00554, and using the feature points, of 0.0082. These results showed the validity of implementing Direct Linear Transformation method for the reconstruction of synthetic scenes, using the essential matrix or the feature points.

However, the extrinsic parameters were not available in the real scenes, and despite neither the intrinsic parameters, they might be computed from the EXIF tags under certain assumptions (see Annex 1).

Real Scenes

As mentioned before, a rough calibration matrix might be computed using the EXIF tags. However, in most of the cases the focal length was modified from one image to other.

Considering these limitations, the scenes were reconstructed from the points using Structure From Motion (SFM), assuming that the camera was not projective. The toolbox of Vincent Rabaud [7] was employed in this section.

Within this toolbox, if the number of frames was equal to 2, the Gold Standard for Affine camera matrix method was implemented to find the projection matrices. If the number of frames was greater than 2, Tomasi-Kanade and autocalibration methods were used.

With this toolbox and using the original set of points extracted from each pair of images, the projection matrices were computed. Afterwards, the 3D scene was reconstructed with the full set of points using Direct Linear Transformation, equation (6), included in the function '`dirLinTransf`'.

The reconstructed scene corresponded to the Cloud of points used in the following section.

Cloud of points matching

The bones reconstructed in the previous step were normalized and matched with their corresponding in the set of meshes. To perform the matching, the algorithm of Iterative Closest Point (ICP) was implemented.

In this case, the ICP toolbox of Wilm and Kjer was used [8]. With this toolbox it was possible to test three different matching methods: 'Brute force', 'Delaunay's' and 'K-D tree'.

Finally, the 3D models were repositioned according to the transformation provided by the toolbox. The sizes of the provided meshes were resized to fit the extracted cloud of points.

Set of images

The provided set of images and their properties are described in the current section. These properties were obtained through the EXIF tags.

The set of images can be divided in two main scenes, as shown in Figure 3:

1. Arrangements of bones of the leg (Scene 1 and 2).
2. Arrangements of several bones at different excavation levels (Scenes 3, 4 and 5).

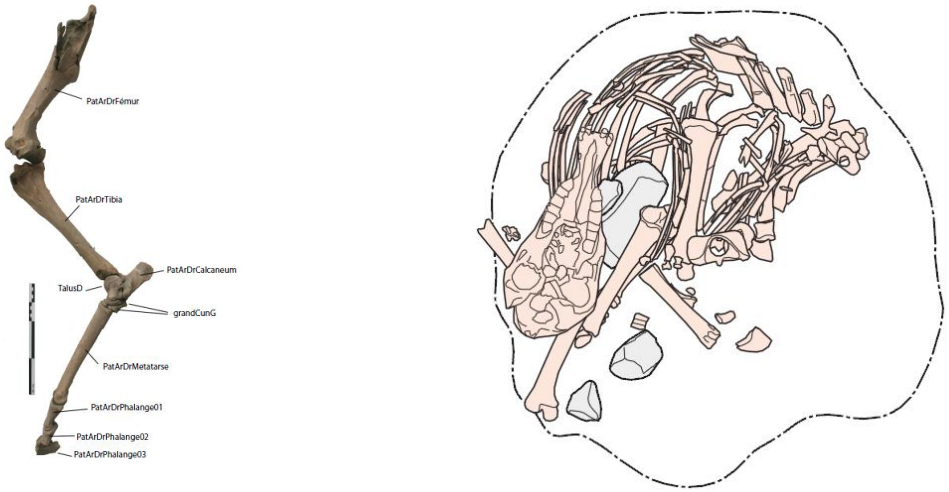

Figure 3. Main scenes: bones of the leg (left) and several bones (right).

Scene 1:

Identified bones: Tibia, calcaneous, talus, metatarsus and phalanxes 1, 2 and 3 (Figure 4).

Acquisition camera: NIKON D70s.

Focal lengths: 44mm and 48mm respectively.

Focal points: f/7.1.

Figure 4. Images of Scene 1.

Scene 2:

Identified bone: Femur (Figure 5).

Acquisition camera: NIKON D70s.

Focal length: 60mm.

Focal points: f/6.3, f/6.3 and f/7.1 respectively.

Figure 5. Images of Scene 2.

Scene 3:

Identified bones: Skull, humerus and pelvis (Figure 6).

Acquisition camera: NIKON D70s.

Focal lengths: 22mm and 18mm respectively.

Focal point: f/5.

Figure 6. Images of Scene 3.

Scene 4:

Identified bones: Skull, jaw, humerus, pelvis and metacarpus (Figure 7).

Acquisition camera: NIKON D70s.

Focal lengths: 62mm and 70mm respectively.

Focal points: f/5.6 and f/6.3 respectively.

Figure 7. Images of Scene 4.

Scene 5:

Identified bones: Skull, tibia and metacarpus (Figure 8).

Acquisition camera: NIKON D70s.

Focal lengths: 58mm and 70mm respectively.

Focal points: f/8.

Figure 8. Images of Scene 5.

Experiments and Results

Below are presented the results obtained at different stages, using the images presented in the previous section.

Feature detection and matching

As mentioned before, the feature points of the images were extracted using FAST, SURF, MSER, Harris corners and Minimum eigenvalues. Later on, the feature vectors were computed with SURF and the matching between the pairs of images was performed. The outliers were eliminated using RANSAC. The inliers for all the scenes can be observed on the left column of Figure 9.

The matching of the additional points generated assuming affine transformations and a close-up of them are presented in the central and right column of Figure 9.

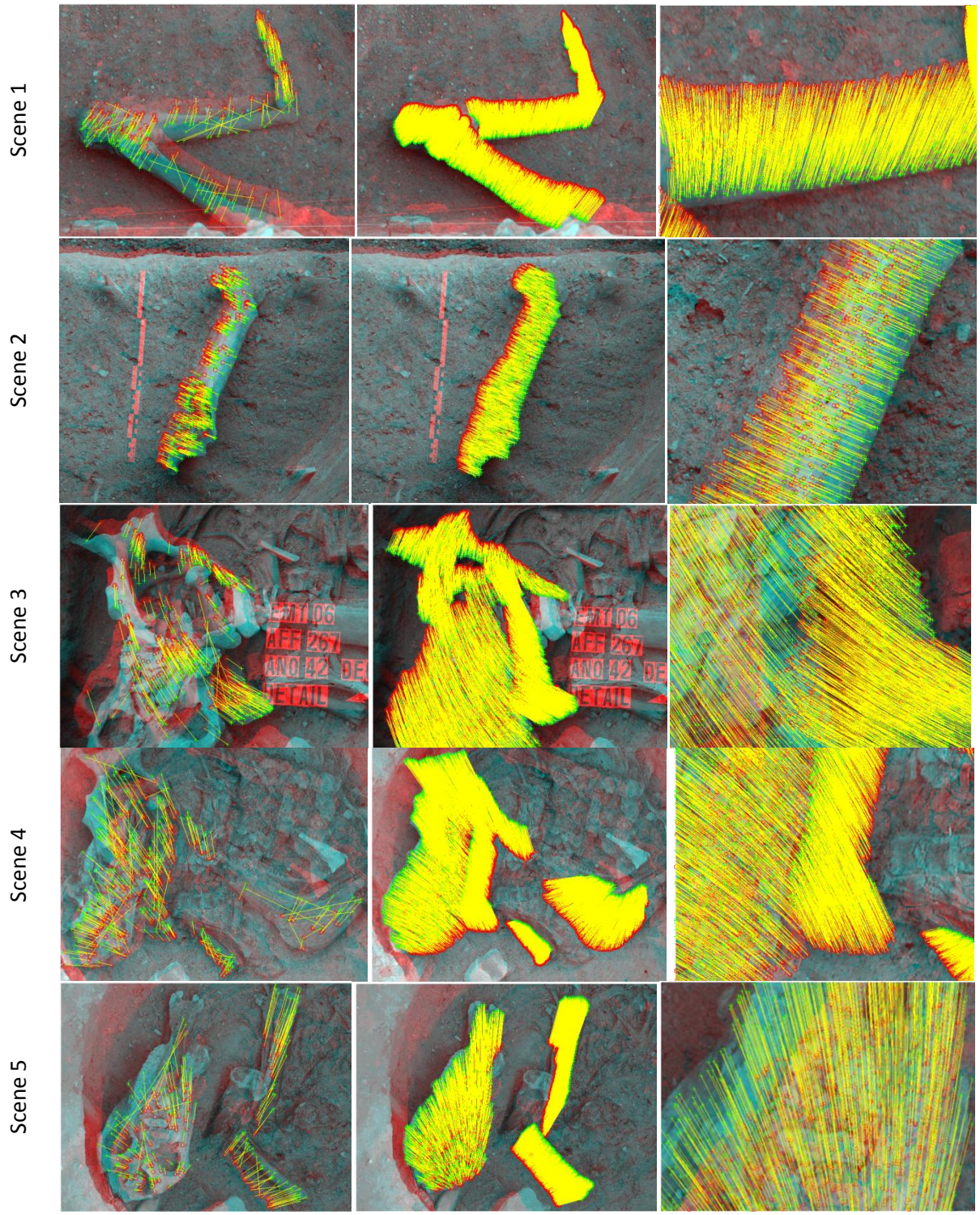

Figure 9. Matching of feature points and additional generated points for each scene: features points (left), generated points (center), close-up of generated points (right).

3D Reconstruction and Cloud of points matching

As presented in the previous section, each pair or set of images were not acquire using the same focal length. This prevented the implementation of the calibration step explained in Annex 1. For that reason, the projection matrices were computed in the SFM toolbox using the methods for uncalibrated cameras.

With the projection matrices, the reconstruction of the 3D scene was performed using Direct Linear Transformation. The cloud of points generated for each bone was matched with the provided 3D model using three algorithms of ICP: 'Brute force', 'Delaunay's' and 'K-D tree'. Afterwards, the set of 3D models were arranged to reconstruct the scenes.

The results can be observed in Figures 10 – 14.

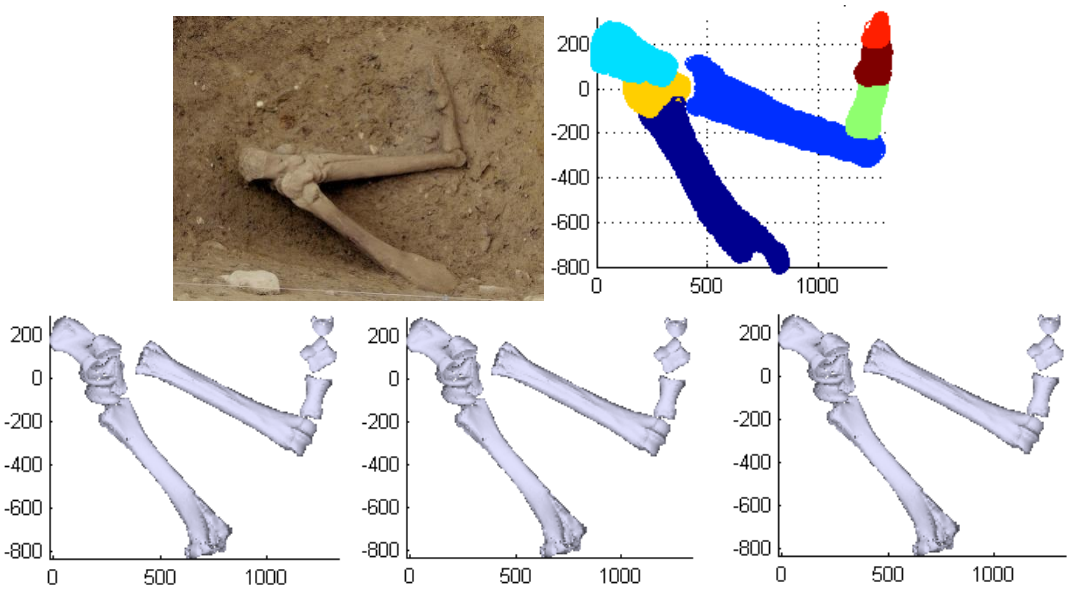

Figure 10. Top: Original image and Scene 1 reconstructed: Dark blue: tibia; Yellow: talus; Cyan: calcaneous; Blue: metatarsus; Green: phalanx 1; Brown: phalanx 2; Red: phalanx 3. Bottom: Cloud of points matching results using: 'Brute force', 'Delaunay's' and 'K-D tree'.

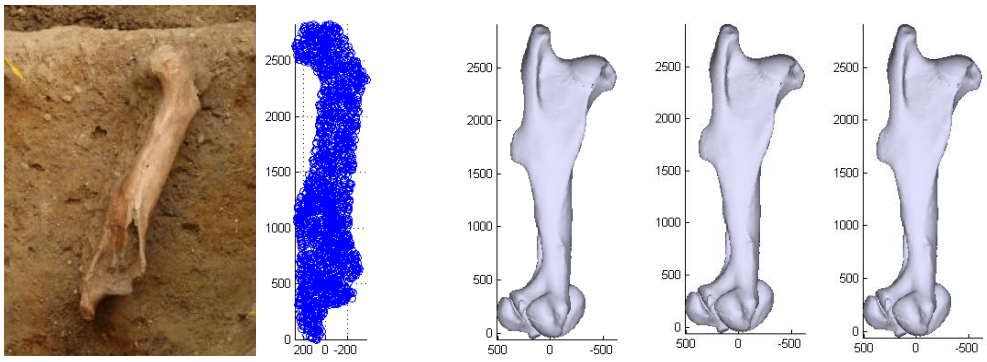

Figure 11. Original image and Scene 2 reconstructed: femur.
Cloud of points matching results using: 'Brute force', 'Delaunay's' and 'K-D tree'.

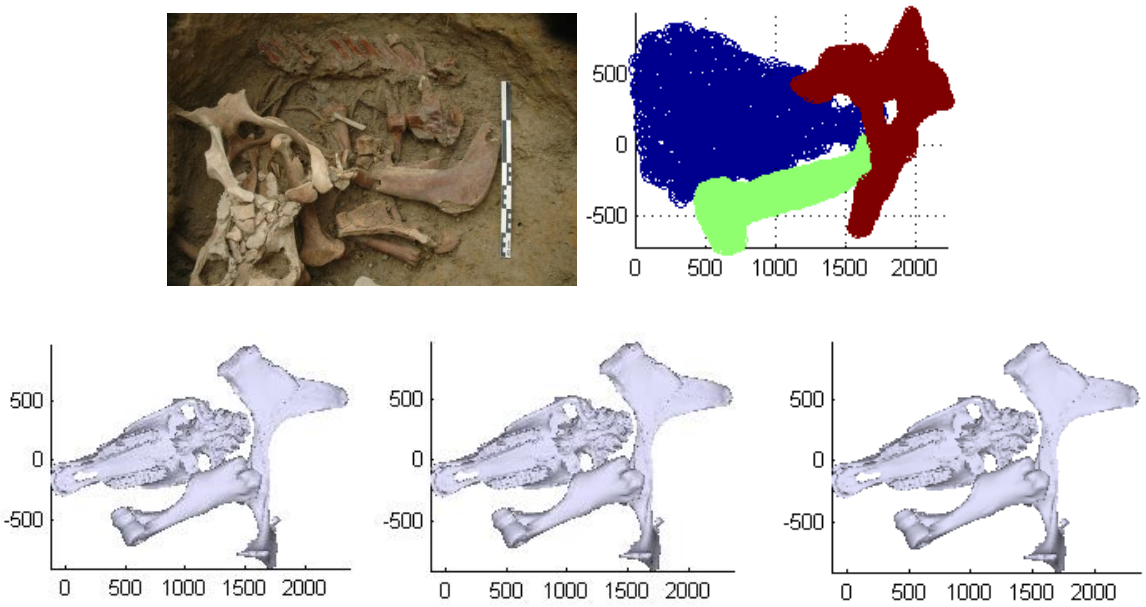

Figure 12. Top: Original image and Scene 3 reconstructed: Blue: skull; Green: humerus; Red: pelvis.
Bottom: Cloud of points matching results using: 'Brute force', 'Delaunay's' and 'K-D tree'.

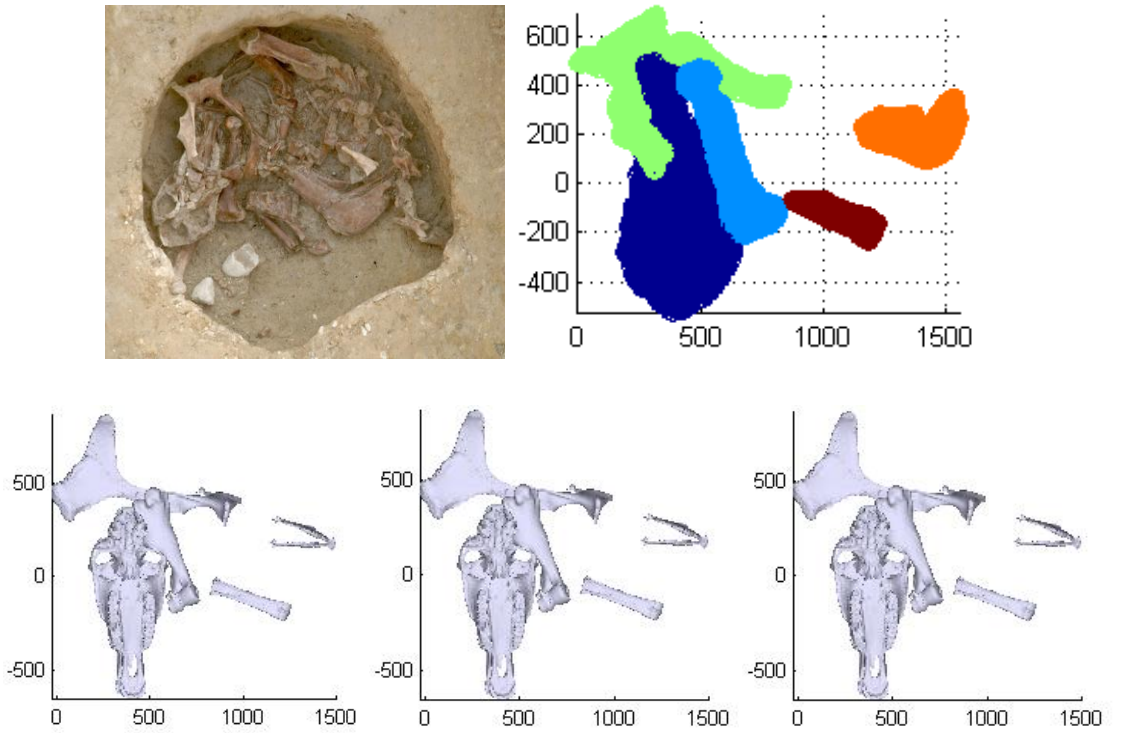

Figure 13. Top: Original image and Scene 4 reconstructed: Dark blue: skull; Blue: humerus; Green: pelvis; Orange: jaw; Brown: metacarpus.
Bottom: Cloud of points matching results using: 'Brute force', 'Delaunay's' and 'K-D tree'.

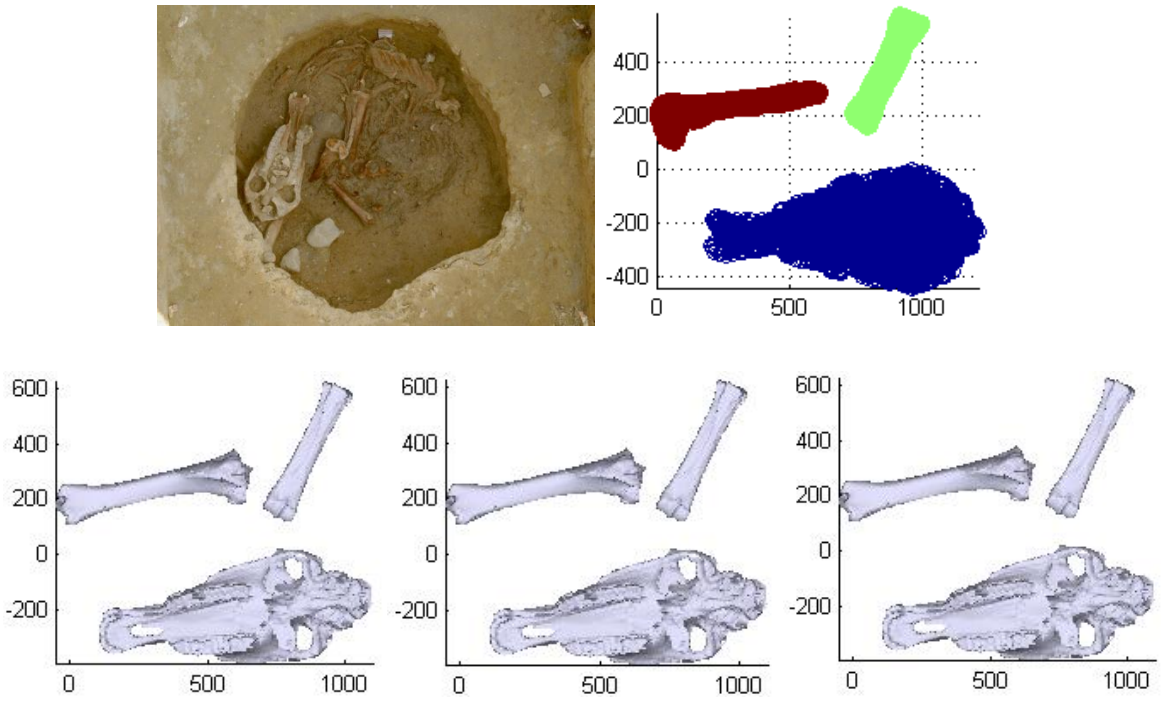

Figure 14. Top: Original image and Scene 5 reconstructed: Blue: skull; Green: metacarpus; Red: tibia. Bottom: Cloud of points matching results using: 'Brute force', 'Delaunay's' and 'K-D tree'.

The execution time in seconds of each ICP algorithm is presented below:

	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5
Brute Force	24.97	16.79	23.99	28.13	23.03
Delaunay's	99.49	56.31	79.98	78.65	78.48
K-D tree	8.22	2.73	4.05	5.25	4.52

Conclusions

In this document, the set of computer vision tools implemented to determine the spatial arrangement of bones of horses during the excavations through a set of images are presented.

One goal was to create cloud of points of the bones, exploiting the information of the images, to relocate the provided 3D models. In order to do so, features points were extracted for each bone using different algorithms, but due to the low number of points to construct a cloud of points, additional points were generated assuming an affine transformation. Despite having some outliers in the feature points, the resulting generated points were accurate.

SFM for uncalibrated cameras was implemented to find the projection matrices and Direct Linear Transformation method was used for the reconstruction. Since the acquired images did not cover many different views from the bones, the real depths of the bones were not visible in the reconstructed ones. This led to the mismatching in the orientation of some bones.

The results obtained after applying three algorithms of ICP: 'Brute force', 'Delaunay's' and 'K-D tree' were the same, despite the execution time varied from one to other, being K-D tree the fastest and Delaunay's the slowest.

References

[1] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *IEEE International Conference on Computer Vision*, vol. 2, 2005, p. 1508–1511.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Computer Vision and Image Understanding (CVIU)*, vol. 110, No. 3, 2008, pp. 346-359.

[3] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *Proceedings of British Machine Vision Conference*, 2002, pp. 384-396.

[4] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, p. 147–151.

[5] J. Shi and C. Tomasi, "Good Features to Track," in *9th IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[6] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.*, Second Edition ed. Cambridge University Press, 2003.

[7] V. Rabaud. Vincent's Structure from Motion Toolbox. [Online]. http://vision.ucsd.edu/~vrabaud/toolbox/

[8] J. Wilm and H. M. Kjer. (2014, Jan.) MATLAB Central. [Online]. http://www.mathworks.com/matlabcentral/fileexchange/27804-iterative-closest-point

[9] N. Snavely. Frequently asked Questions (FAQ) about Bundler. [Online]. http://www.cs.cornell.edu/~snavely/bundler/faq.html

[10] G. Roth. Camera Calibration. [Online]. http://people.scs.carleton.ca/~roth/comp4900d-12/notes/lect12_camera.pdf

[11] P. Cignoni. VCG Library. [Online]. http://vcg.isti.cnr.it/~cignoni/newvcglib/html/shot.html

[12] N. Snavely. (2008, Jan.) Estimating the focal length of a photo from EXIF tags. [Online]. http://phototour.cs.washington.edu/focal.html

Annex 1.

Calibration: Intrinsic Parameters

In the pinhole camera model, the intrinsic matrix is given by (1).

$$K = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

If the calibration of the camera is not available, it is possible to make use of the EXIF tags to obtain an estimation of the parameters of *K* [9], [10] and [11]. Two cases are listed below:

Case 1:

Information available

- EXIF tags.

Assumptions

- Principal point (u_0, v_0) is assumed to be in the center of the image.
- Skew coefficient γ is assumed to be 0.
- Nonlinear intrinsic parameters are not considered.

When the camera is not calibrated but the files contain EXIF tags, the calibration matrix can be obtained as follows:

1. EXIF tags are extracted from an image file, specifically 'focal length' and 'width of the image sensor' (CCD Width). Usually the last one is not available in the file, but it can be found on the internet using the reference of the camera.
2. Estimate the focal length of a photo, using equation in (2) [12]:

$$\begin{aligned} f_x &= (\text{image width in pixels}) * (\text{focal length in mm}) \,/\, (\text{CCD width in mm}) \\ f_y &= (\text{image height in pixels}) * (\text{focal length in mm}) \,/ (\text{CCD height in mm}) \end{aligned} \quad (2)$$

   Where f_x and f_y are the focal length in pixels in the x and y coordinates.
3. Fill the intrinsic matrix.

Case 2:

Information available

- EXIF tags.
- Fundamental matrices of several pair of images.

Assumptions

- Principal point (u_0, v_0) is assumed to be in the center of the image.
- Skew coefficient γ is assumed to be 0.
- Nonlinear intrinsic parameters are not considered.

If both EXIF tags and the fundamental matrices of several pair of images are available, it is possible to use an autocalibration algorithm to obtain a more accurate intrinsic matrix. The prior calibration matrix is obtained using the previous case steps (files: intrinsicParFromEXIF.m and autoCamCal.m).