Horizon 2020
European Union funding
for Research & Innovation

ITS 4 LAND
innovations for land tenure

Deliverable 3.5

Technical Report

1st August 2018

Version 1.0

Abstract:
Report on the implementation of a matching algorithm for the alignment of sketch maps with corresponding cartographic maps

Project Number: 687828
Work Package: 3
Lead: WWU
Type: Other
Dissemination: Public
Delivery Date: 1st August 2018
Contributors: Malumbo Chipofya, Sahib Jan, Cristhian Murcia, Angela Schwering, Carl Schultz, Mina Karamesouti, Stephanie Walter, Humayun, Mohammed, Christian Timm

Its 4 Land
Hengelosestraat 99
Enschede 7500AE

Netherlands
Phone: +31534874532
www.its4land.com

The its4land consortium consists of the following partners:

University of Twente (UT)
KU Leuven (KUL)
Westfaelische Wilhelms-Universitaet Muenster (WWU)
Hansa Luftbild AG (HL)
Institut d'Enseignement Superieur de Ruhengeri (INES)
Bahir Dar University (BDU)
Technical University of Kenya (TUK)
ESRI Rwanda (ESRI).

Executive Summary

This document reports the outcomes of work leading to deliverable D3.5 under the task T3.3 “Sketch-to-Geo”. Deliverable 3.5 is the fifth deliverable of work package WP3, Draw and Make. D3.5 reports on the approach used for qualitative alignment of sketched information with underlying geo-referenced datasets. In particular, D3.5 demonstrates the applicability of robust matching algorithms for the sketch map alignment problem.

Qualitative alignment is the process of matching the qualitative descriptions of a pair of spatial scenes. It involves searching for a correspondence between objects in one scene with objects in the other such that the similarity of spatial relations between corresponding pairs is maximized. The qualitative descriptions are fundamental to these processes because the measure of similarity depends on the mutual agreement between matched pairs across the qualitative descriptions. The qualitative descriptions of the maps involved are computed using the qualifier tool reported in deliverable D3.3. The qualifier takes as input a spatial configuration of geographic features and produces qualitative spatial scene description consisting of a set of graphs called Qualitative Constraint Networks (QCNs). QCNs are graphs where the nodes represent geometric features and the edges represent spatial relations between them.

The tool presented in this report takes as input as a pair of qualitative spatial scene descriptions and applies an approximate dynamic programming that uses a default search policy based on our *local compatibility matrix* model.

Its 4 Land
Hengelosestraat 99
Enschede 7500AE
Netherlands
Phone: +31534874532
www.its4land.com

Contents

Abbreviations

QCN	Qualitative Constraints Network
LCM	Local Compatibility Matrix

1. Introduction

Its4land is a European Commission Horizon 2020 project funded under its Industrial Leadership program, specifically the 'Leadership in enabling and industrial technologies – Information and Communication Technologies ICT (H2020-EU.2.1.1.)', under the call H2020-ICT-2015 – and the specific topic – 'International partnership building in low and middle-income countries' ICT-39-2015.

Its4land aims to deliver an innovative suite of land tenure recording tools that respond to sub Saharan Africa's immense challenge to rapidly and cheaply map millions of unrecognized land rights in the region. ICT innovation is intended to play a key role. Many existing ICT-based approaches to land tenure recording in the region have failed: disputes abound, investment is impeded, and the community's poorest lose out. its4land seeks to reinforce strategic collaboration between the EU and East Africa via a scalable and transferrable ICT solution. Established local, national, and international partnerships seek to drive the project results beyond R&D into the commercial realm. its4land combines an innovation process with emerging geospatial technologies, including smart sketch maps, UAVs, automated feature extraction, and geocloud services, to deliver land recording services that are end-user responsive, market driven, and fit-for-purpose. The transdisciplinary work also develops supportive models for governance, capacity development, and business capitalization. Gender sensitive analysis and design is also incorporated. Set in the East African development hotbeds of Rwanda, Kenya, and Ethiopia, its4land falls within TRL 5-7: 3 major phases host 8 work packages that enable contextualization, design, and eventual land sector transformation. In line with Living Labs thinking, localized pilots and demonstrations are embedded in the design process. The experienced consortium is multi-sectorial, multi-national, and multidisciplinary. It includes SMEs and researchers from 3 EU countries and 3 East African countries: the necessary complementary skills and expertise is delivered. Responses to the range of barriers are prepared: strong networks across East Africa are key in mitigation. The tailored project management plan ensures clear milestones and deliverables, and supports result dissemination and exploitation: specific work packages and roles focus on the latter.

This document reports on deliverable D3.5 which is directly linked to the task T3.3 "sketch-to-geo" of the work package (WP3) in the project. WP3 aims at developing a software tool, Smart SkeMa (pronounced smärt skē-mə) in short for recording land tenure information based on hand-drawn sketch maps. Our goal is to fill the gap in this area that has been left by traditional GIS systems. Smart SkeMa is poised to do so by providing the means to automatically digitize hand drawn maps, geo-localize the main elements in the maps using an existing base map as a reference, and providing the means to visualize and further annotate the maps with relevant concepts. The tool is composed of several components including a model of land use concepts, recognition and extraction of objects in sketch maps, qualitative representation of input maps, and qualitative alignment of sketched information with

underlying geo-referenced datasets. All these components come together to provide a single function: integrating the user's sketch into a base topographic dataset.

The integration of sketch information requires geolocalization of sketched objects: a process by which objects drawn in a freehand sketch are grounded within an underlying geo-referenced dataset. It involves finding correspondences between the spatial entities in the first scene and those in the second that, in a sense, respect the spatial relations within the scenes being matched [1]. For this task object annotations and spatial relations can be seen as constraints on a configuration so that finding the best alignment becomes finding a correspondence between objects that minimizes constraint violations, i.e. a graph matching problem. This can be done by listing all the nodes from sketch map graph and inserting a labelled arc between any two nodes connected by an arc in the original graph. The alignment process can then be thought of as the task of similarly listing the metric map nodes and then permuting them so that the first n nodes in the metric map list together with their out-going arcs correspond to the first n nodes of the sketch map list together with their out-going arcs in a way that minimizes the differences between corresponding constraints.

In T3.3 we have developed a graph matching algorithm that performs this task. The algorithm takes qualitative spatial representations of a sketch map and a geo-referenced map as input. Qualitative spatial representations involve representing only the relevant distinctions in a spatial configuration using some form of qualitative relations such as *left_of, right_of, near,* and *far.* In the deliverable D3.3, we implemented a qualifier (a software tool for computing qualitative representations of maps). The qualifier takes vector representations of maps as input and generates graph representations known as Qualitative Constraint Networks (QCNs) from the spatial configurations. The matching algorithm uses these QCNs as input for the alignment of spatial objects from sketch maps with corresponding geo-referenced maps.

The algorithm presented in this report has been implemented in python and is submitted as part of the Demo accompanied by this report. For demonstration purposes, the qualifier and matching algorithm have been integrated with a web-based user interface[1] (see Appendix 1 for usage guidelines).

The remainder of this report is structured as follows. Section 2 gives overview on qualitative representation of sketch maps with an example using the spatial relations Left and Right. Section 3 describes the graph matching algorithm developed in T3.3 for the qualitative alignment process and Section 4 concludes the report on deliverable D3.5.

2. Qualitative Representation of Input Maps

[1] https://share4land.itc.utwente.nl:5566/sharing/eoaWuDHjv

Qualitative representation of maps involves representing only the relevant distinctions in a spatial configuration using qualitative relations. In the area of qualitative spatial reasoning (QSR) dozens of representational languages, more commonly called qualitative spatial calculi, have been proposed. These calculi formalize spatial configurations using relations over the set of spatial entities. For the qualitative representation of input maps, we have implemented a qualifier (deliverable D3.3). The qualifier contains a set of modules representing spatial relations of the spatial calculi. Each module represents spatial aspects such a topology, relative orientation, ordering and distances. For each spatial aspect, the implemented qualifier formalizes the spatial configurations of the input maps as QCNs. The implemented spatial aspects are: (i) topological relations between polygonal features**,** (ii) topological relations between spatial features of different dimensions, (iii) topological relations between linear features, (iv) relative orientation of polygonal features, (v) relative orientation of spatial features of different dimensions, (vi) relative orientation of linear features, (vii) linear ordering of polygonal features, (viii) relative distances between polygonal features.

The qualitative representation requires geometric representation of input maps. In our previous deliverable D3.2, we have demonstrated how our sketch recognition component extracts and interprets the drawn object into meaningful geometric entities such as points, lines, and polygons. Figure 1 shows a real sketch map example, drawn by a member of the Maasai community during one of our field visit in Kenya (early, 2017). The spatial objects in the drawn map are automatically extracted using the advanced recognition methods presented in [3]. Figure 2 shows the QCNs representing the Left/Right relations between spatial objects with respect to rivers in the input maps and possible matches of spatial objects across the input maps. These QCNs are use as input for the alignment task.

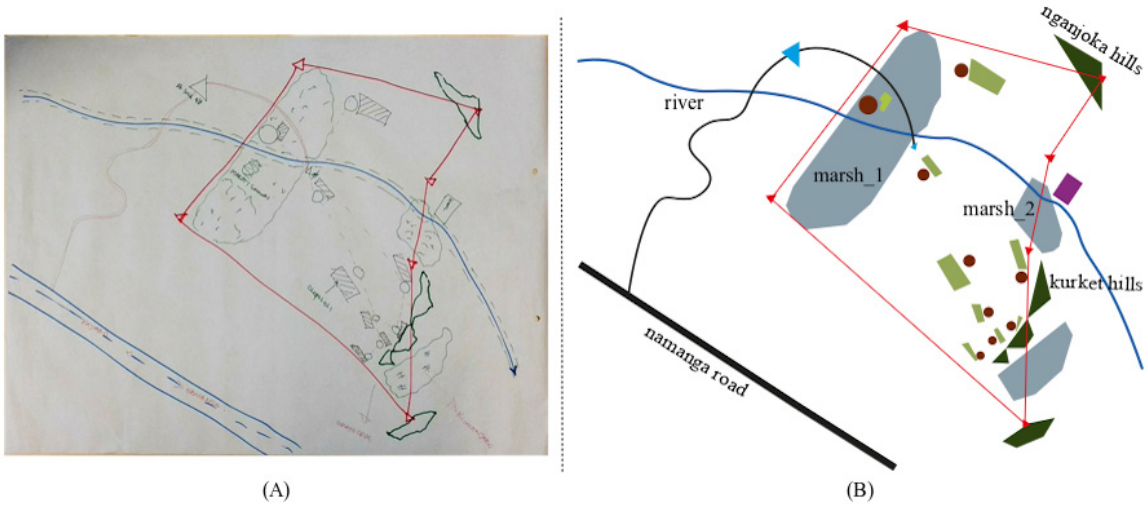

Figure 1. (A) sketch map drawn by community members, (B) Vector representation of sketch map using implemented recognition methods in the deliverable D3.2.

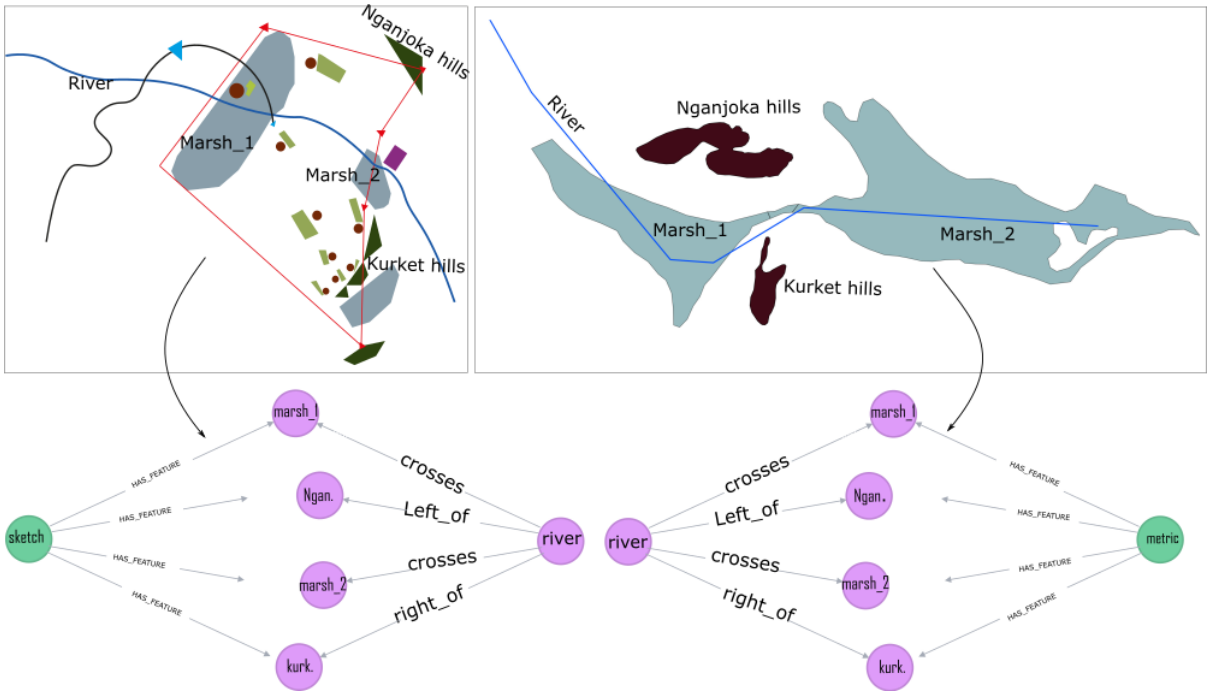

Figure 2. Vectorized sketch map, corresponding geo-referenced map and qualitative representation of input maps as QCNs using the LeftRight relations (left_of, right_of, and crosses).

3. Graph Matching Algorithm

The matching algorithm used by Smart SkeMa is based on our previous work on sketch map alignment reported in [1, 2]. Key to our current approach is the use of relation similarities to measure how well any pair of features fit together and the computation of a heuristic $\mathbf{e}_{(i,j)|\mathbf{m}}$, based on a model we call the *local compatibility matrix* (LCM) model, to explore the space of potentially matching pairs of features, one each from the metric map and sketch map. Here **m** is the current match and **(i, j)** are a candidate pair to be added to **m** as explained further below.

The high-level workflow of our approach proceeds as follows: first map features in both maps are compared for type similarity. Similar features are paired up to form a subset of the cross product of the sets of features from the two maps. The paired features are then compared for relation similarity. The similarity values thus computed are then tabulated into a matrix indexed on both dimensions by the set of paired features. We refer to this matrix as the similarity matrix for the input maps – in the literature (e.g. [7]) this is also referred to as the compatibility matrix. The similarity matrix then becomes the input to our matching algorithm. We considered two approaches for the matching algorithm design. The first is Leordeanu and Herberts [6] spectral graph matching approach based on using the principal

eigenvector of the similarity matrix as a global heuristic evaluation function in their algorithm. The second approach which we describe in more detail below uses the similarity values to compute local heuristic estimates $\mathbf{e}_{(i,j)|m}$ and to update a surrogate value function that may learn better values of the pairs being matched in the course of executing the algorithm. In the remainder of this section, we describe these components of our approach in more detail.

Encoding map similarity

There exist many approaches to compute feature similarity based on their types and spatial relations (see e.g. Chapter 2 of [8] for an overview). In general, both types and relations may be viewed as concepts in a domain model. As in most approaches in the literature, we compute similarities based on a notion of concept or relation distances. The similarity **s** is derived from the distance **d** as **s = 1 – d**.

Feature type similarity

There are two approaches for computing concept distances. In the simplest case, two concepts either match exactly in which case their distance is **0** or they do not, in which case it is **1**. The second approach to distance computation is based on the concept hierarchy in our domain model. We use Wu and Palmer's the simple distance measure [5] between two concepts. If **a** and **b** are two concepts in a concept hierarchy and **LCA(a,b)** denotes their least common ancestor in the hierarchy and **h(a)** is the height (or depth) of the concept **a**, then their distance is given by

$$\mathbf{d(a,b) = (h(a)+h(b) – 2 \cdot h[LCA(a,b)])/ (h(a)+h(b))}$$

The similarity values computed using **d(a,b)** will be in the interval [0, 1). To pair up features we must use a threshold such that all pairs whose similarity is larger than or equal to the threshold are paired as similar features.

Relation similarity

Relation similarity is computed for each pair among the features paired using the feature similarity measure. Suppose $\mathbf{a_s}$ and $\mathbf{b_s}$ are features in the sketch map and $\mathbf{a_m}$ and $\mathbf{b_m}$ are features in the metric map such that $\mathbf{(a_s, a_m)}$ and $\mathbf{(b_s, b_m)}$ have be respectively paired up by feature type similar. Then for each calculus for which both $\mathbf{(a_s, b_s)}$ and $\mathbf{(a_m, b_m)}$ have a relation specified, the distance between the *matches* $\mathbf{(a_s, a_m)}$ and $\mathbf{(b_s, b_m)}$ is given as the conceptual neighborhood distance in that calculus between the relations $\mathbf{R(a_s, a_m)}$ and $\mathbf{R(b_s, b_m)}$. For more on conceptual neighborhoods for qualitative spatial relations we refer the reader to [4, 9].

Once we have computed the similarity of all relations between a pair of matches e.g. $\mathbf{(a_s, a_m)}$ and $\mathbf{(b_s, b_m)}$ we aggregate them by taking their weighted average using pre-assigned weights for each calculus.

The similarity matrix

To tabulate the data into the similarity matrix we index each pair $\mathbf{(a_s, a_m)}$ by $\mathbf{a = a_s \cdot size_m + a_m}$, where $\mathbf{size_m}$ is the size of the metric map. The matrix entry at then represents the relation similarity of pair **(a, b)** b if $\mathbf{a \neq b}$ or the type similarity of $\mathbf{(a_s, a_m)}$ if $\mathbf{a=b}$.

LCM based heuristic for graph matching

Given a similarity matrix M with real values, one can obtain a binary matrix by applying a threshold over the M sending all values below the threshold to **0** and the remainder to **1**. Commonly referred to as a compatibility matrix this specifies for each possible combination of features pairs from the two maps, how well the pairs fit together given the similarity threshold. By summing up the number of ones in a row of the compatibility matrix can give an indication of the number of other pairs that are compatible with the pair for that row.

We refine the idea of the compatibility matrix by decomposing the matrix into individual rows and reshaping the row into a $\mathbf{size_s \times size_m}$ matrix where each row represents a sketch map feature and each column represents a metric map feature. If $\mathbf{a = a_s \cdot size_m + a_m}$ is index of the row from which the matrix was derived, then the entry at the intersection of row $\mathbf{b_s}$ and column $\mathbf{b_m}$ has is the value of the entry at $\mathbf{b = b_s \cdot size_m + b_m}$ in the original compatibility matrix. We call this new, derived, matrix a Local Compatibility Matrix (LCM) and refer to the pair $\mathbf{(a_s, a_m)}$ as its reference pair.

The power of the LCM comes in that it allows us to efficiently determine an upper bound on the size of any solution (i.e. set of consistent matches) that can include the reference pair of the LCM. We denote this upper bound $\mathbf{e_{(i,j)|m}}$ and use it as an estimate of the value of using the associated pair as candidate to be included in the final set of matches. For more details about LCMs and the determination of the values $\mathbf{e_{(i,j)|m}}$ we refer the reader to [1, 2]. In the next section below we show how $\mathbf{e_{(i,j)|m}}$ is used as a heuristic in our matching algorithm.

Heuristic graph matching with approximate dynamic programming

The matching algorithm we have designed for D3.5 is conceptually simple. Given the set of all candidate pairs (potential matches) pick one with a high value, add it to the current solution, and repeat until there are no more pairs available. The function executing the algorithm expects to receive a maximum number of iterations for which it must repeat this

process each time creating a solution and comparing it with the best previously found solution.

The problem with this hill-climbing technique is that it is possible to get stuck in bad regions of the search space since the estimates used may be overly optimistic. To overcome this problem we use a so-called ϵ-greedy strategy to pick the next candidate to use. In this strategy the best available candidate is chosen in every step except with a small probability ϵ a random candidate is chosen and used. While this ensures alternative paths in search space are explored every once-in-while, it does provide a way for the algorithm to track how good its choices have been up to the current iteration it is executing. We incorporate this capability by employing a dynamic programming architecture.

In approximate dynamic programming [10] a problem is formulated in terms of a system of states, where each state has some associated value. The system will transition between states in response to actions taken (by a purported agent) in an initial state and arrive in a final state at the end of the transition. The goal of the algorithm is to learn to take actions that lead to high value states. It learns this by observing a reward after taking each action and using the observed reward and the value of resulting state to update its perceived value of being in the state from which it took the action.

We formulate our dynamic program as follows. Each feature in the sketch map will represent a level. A state is a pair $\mathbf{(a, } l\mathbf{)}$ where **a** is a candidate pair and l is a level. From any state, $\mathbf{(a, } l\mathbf{)}$ say, the algorithm can perform an add action, which adds an eligible candidate, say **b**, to the current solution to arrive in the new state $\mathbf{(b, } l\mathbf{+1)}$. The reward observed during such a move is

$\mathbf{C(a, } l\mathbf{)} \coloneqq$ the sum of average relation similarities of pairs in the current solution.

In fact the reward the cumulative sum of average similarities computed after each action. With this reward we then apply the dynamic programming update

$$\nu^{n}(\mathbf{a}, l) \leftarrow \mathbf{C(a,}l\mathbf{)} + \gamma \cdot \Sigma\, \mathbb{P}(\mathbf{b})[\mathbf{V}^{n-1}(\mathbf{b}, l\mathbf{+1}) - l] \,,$$

and let

$$\mathbf{V}^{n}(\mathbf{a}, l) = (1 - \alpha)\, \mathbf{V}^{n-1}(\mathbf{a}, l) + \alpha \cdot \nu^{n}(\mathbf{a}, l)$$ for all states $\mathbf{(a, } l\mathbf{)}$ visited in the current iteration.

α is called the stepsize and is used for smoothing the noise in the approximation ν^{n} above. The values are indexed by the superscript **n** to indicate the iteration in which they are encountered. Because the order in which candidates are added does not matter, we propagate the values update backwards across the levels as in

$$\mathbf{V}^{n}(\mathbf{a}, t) = (1 - \alpha^{l-t+1})\, \mathbf{V}^{n-1}(\mathbf{a}, l) + \alpha^{l-t+1} \cdot \nu^{n}(\mathbf{a}, l)$$ for all levels $t \leq l$.

$\mathbf{V^n(a, l)}$ is initialized to $\mathbf{e_{a|\emptyset}}$ for all pairs **a** and levels l. With these structures set up the algorithm now proceeds to iteratively create solutions using the original heuristic as before but every k^{th} iteration it uses the learned values instead to see if it can find a better solution using the learned values.

4. Conclusion

Sketch map alignment is achieved by applying graph matching methods. During the matching both the feature types and spatial relations perform a vital role. The alignment of certain spatial objects (e.g. mountains, rivers, roads, etc.) become anchoring positions to relate other sketched objects and to integrate additionally sketched information into the metric map.

While the algorithm described in this report performs well in the test cases explored so far, it requires correct settings for several parameters if the learned are to be profitably exploited. In simple cases, it does not require consulting the learned values but may benefit in long search runs such as those that can be performed on its4land's platform being developed under WP6.

It is worth noting that the choice of spatial representation has a severe impact on the outcome of map alignment. This is because the inherent schematic structure of sketch maps introduces inconsistent, albeit, systematic distortions. The algorithm takes the results from deliverable D3.3 as input and produces a *correspondence-mapping* from the sketch map to the metric map. In the context of its4land the metric map is a topographic map with both natural and man-made features relevant for community-based land tenure documentation.

Bibliography

[1] M. Chipofya, C. Schultz, and A. Schwering, “A metaheuristic approach for efficient and effective sketch-to-metric map alignment,” *Int. J. Geogr. Inf. Sci.*, vol. 30, no. 2, pp. 1–26, 2016.

[2] M. Chipofya, “Matching Qualitative Constraint Networks with Online Reinforcement Learning,” In: Christoph Benzmüller, Geoff Sutcliffe and Raul Rojas (eds). GCAI 2016. 2nd Global Conference on Artificial Intelligence, vol 41, pages 266--279

[3] M. Chipofya, C. Murcia Galeano, M. Karamesouti, S. Jan, A. Schwering, C. Schultz, “Smart Sketch Maps : A Tool for Community-driven Land Documentation using Hand Drawn Maps,” Masterclass at *World bank Conference on Land and Poverty Conference 2018: Land Governance in an Interconnected World , March 19-23*, 2018.

[4] J. F. Allen, “Maintaining Knowledge about Temporal Intervals,” *Commun. ACM*, vol. 26, no. 11, pp. 832–843, 1983.

[5] Z. B. Wu and M. Palmer, “Verbs semantics and lexical selection,” In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics, 1994. 133-138

[6] M. Leordeanu and M. Hebert, “A spectral technique for correspondence problem using pairwise constraint,” 10th IEEE International Conference on Computer Vision (ICCV’05): 1482–1489, 2005.

[7] Timothee Cour, Praveen Srinivasan, and Jianbo Shi. “Balanced graph matching,” *Advances in Neural Information Processing Systems*, 19 (313), 2007.

[8] Konstantinos A. Nedas. “Semantic Similarity of Spatial Scenes,” PhD thesis, University of Maine, August 2006.

[9] Christian Freksa. “Conceptual Neighborhood and its role in temporal and spatial reasoning,” In: Singh, M., Travé-Massuyès, L. (eds.) Proc. of the IMACS Workshop on Decision Support Systems and Qualitative Reasoning, 181-187, North-Holland, Amsterdam, 1991.

[10] Powell, W., B., “Approximate dynamic programming : solving the curses of dimensionality,” 2nd ed. John Wiley & Sons. Hoboken, New Jersey. 2011.

Appendix 1: Web-based User Interface

The SmartSkeMa system is composed of several components including a system for automated recognition and extraction of sketched objects, qualitative representation, and qualitative alignment of sketched information. All these components come together to provide a single function: integrating the user's sketch into a base topographic dataset.

In order to demonstrate the functionality of three components (sketch recognition, qualitative representation, and alignment), we have implemented a prototype, a web-based user interface. The interface takes sketch and geo-referenced maps as an input, processes sketch map (D3.2), qualifies input maps (D3.3) and aligns the sketch information (D3.5). The object recognition component D3.2 is not fulling integrated in the web-interface. However, we used its output in the web-interface to demonstrate the workflow.

Step 1: Load maps

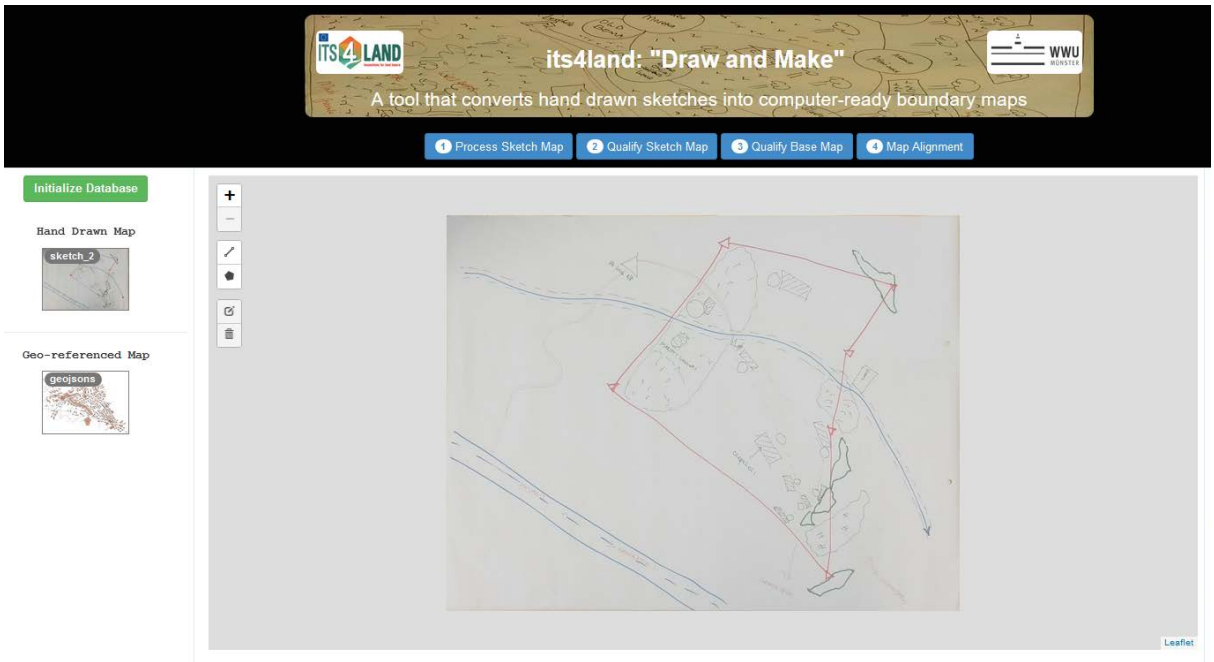

Figure 3. Web-interface for loading sketch and corresponding geo-referenced maps.

s

Step 2: Process sketch map

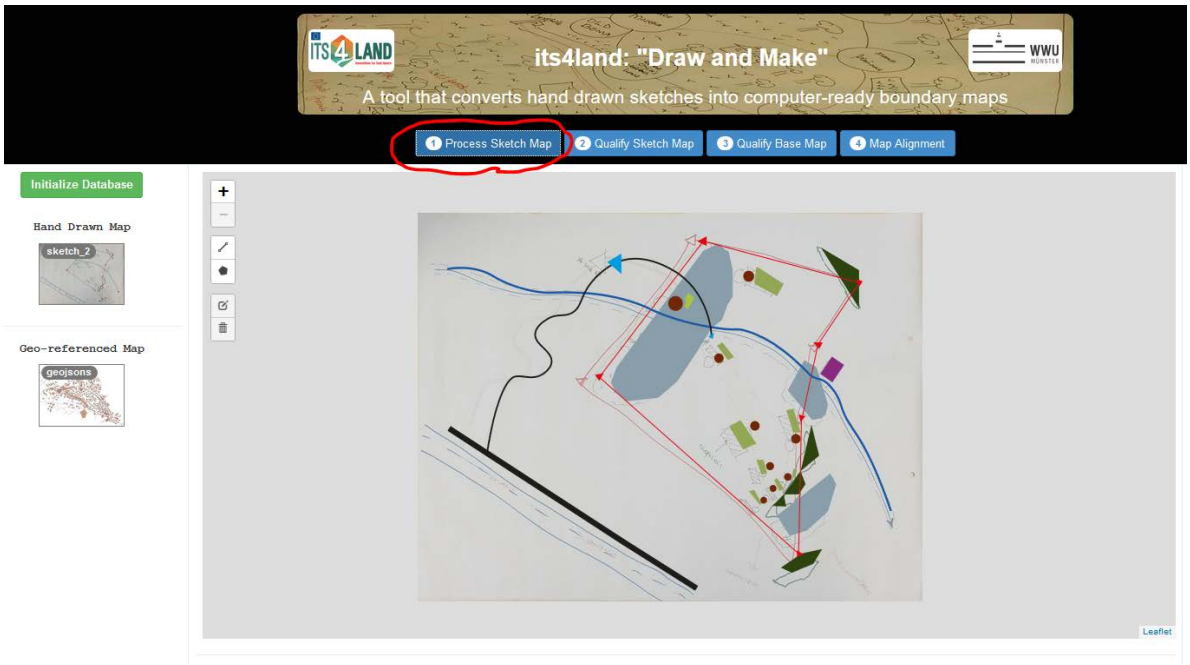

Figure 4. The process recognizes and extracts drawn objects in sketch map and represent them as a vector data.

Step 3: Qualify sketch map

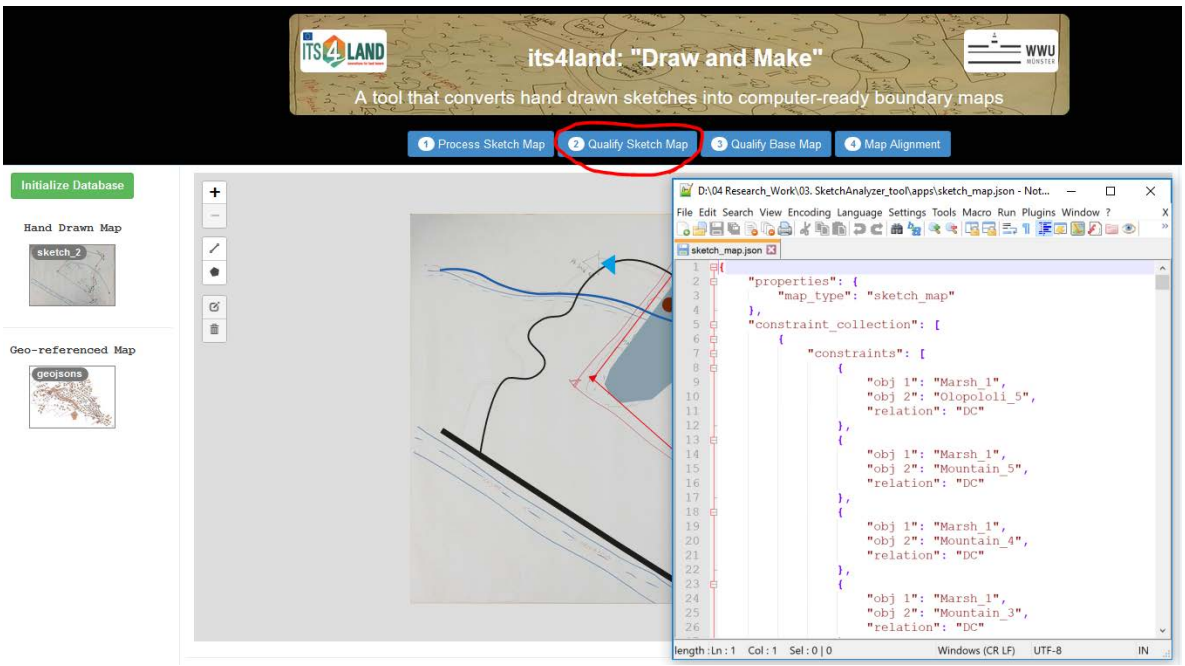

Figure 5. The process takes extracted objects in sketch map and generates QCNs along with other attributes of the geometries in a standard (*.json) format.

Step 4: Qualify metric map

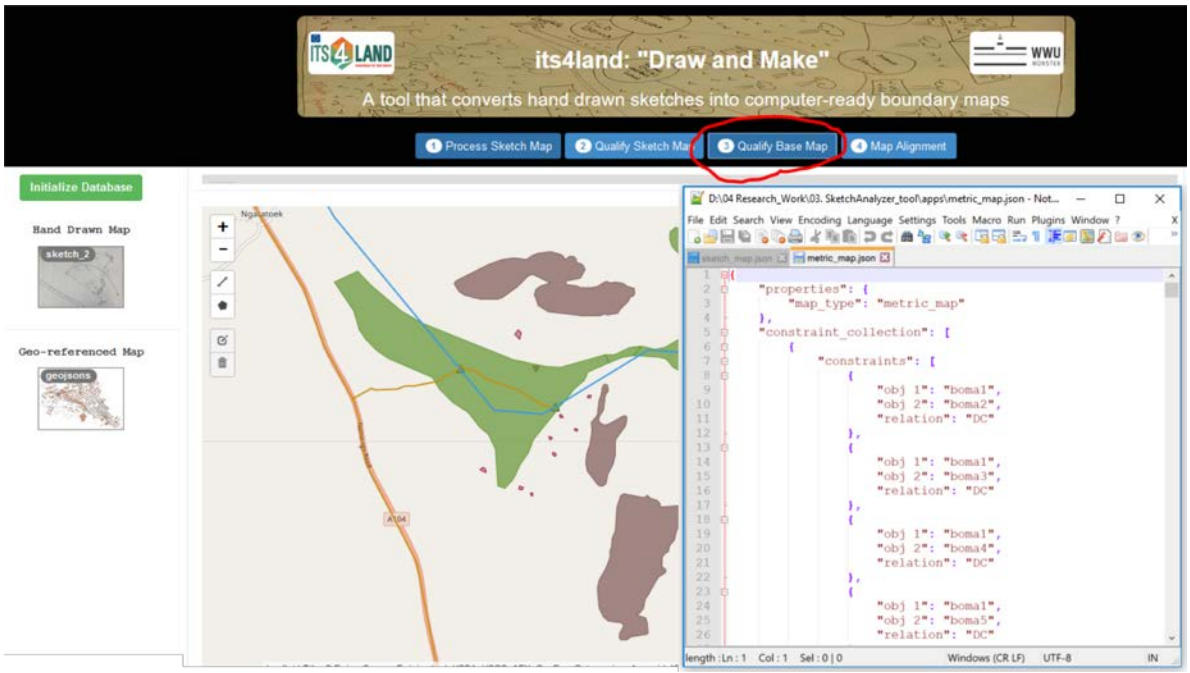

Figure 6. The process takes geo-reference map as an input and generates QCNs along with other attributes of the geometries in a *.json) format.

Step 5: Qualitative alignment of drawn features

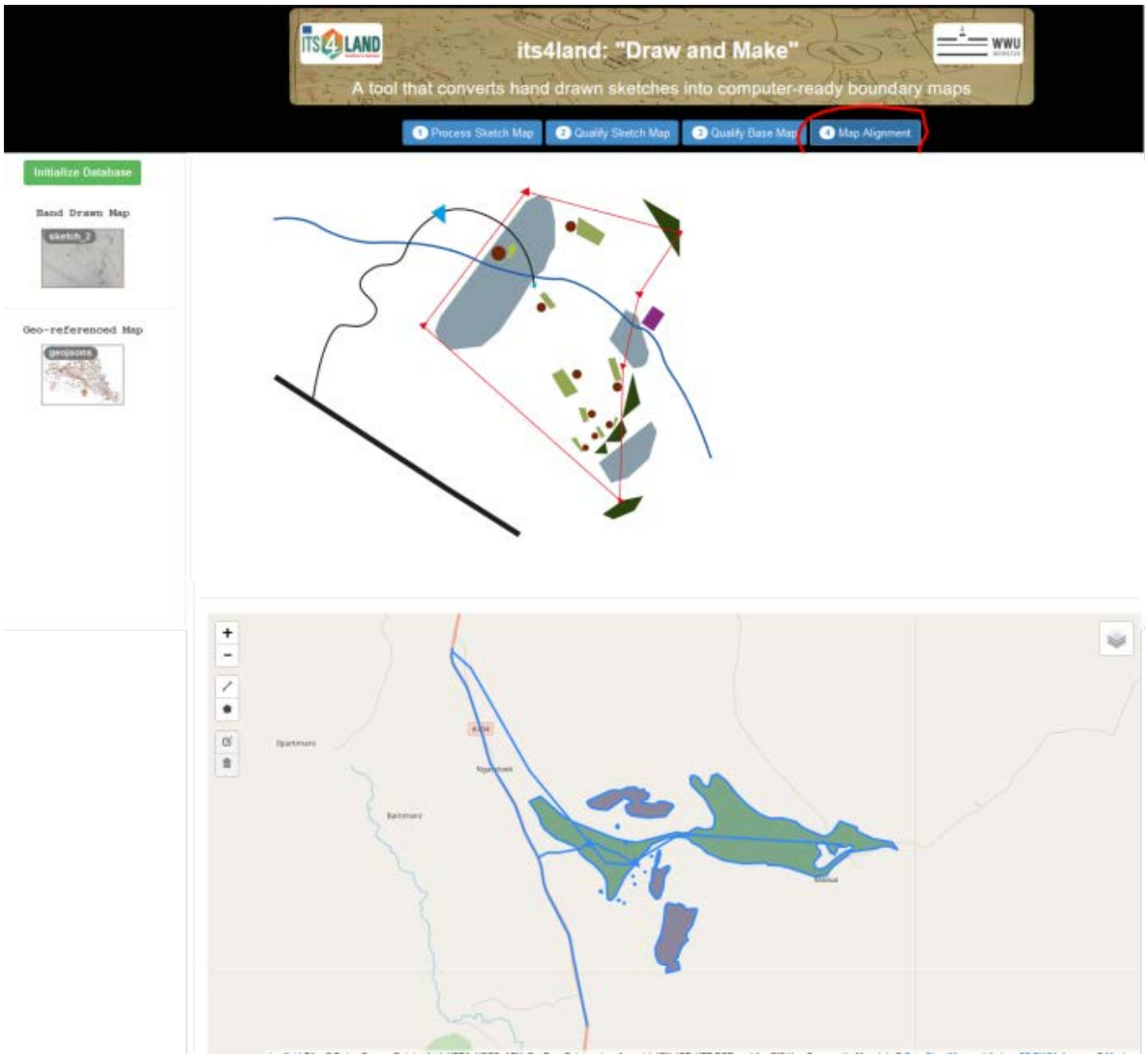

Figure 3. The process aligns spatial objects from sketch map with corresponding object in the geo-referenced map. The interface allow user to interact with aligned objects by mouse click events. When the user clicked on object in sketch map, the corresponding object in the geo-referenced map will highlight.