Semantic-aware spatio-temporal Alignment of Natural Outdoor Surveys

Georges Chahine, Cédric Pradalier

Semantic-aware spatio-temporal Alignment of Natural Outdoor Surveys

Georges Chahine*
College of Engineering
Georgia Institute of Technology, Atlanta, GA 30332
IRL 2958 GT-CNRS - GeorgiaTech Lorraine
gchahine@gatech.edu

Cédric Pradalier
College of Computing
Georgia Institute of Technology, Atlanta, GA 30332
IRL 2958 GT-CNRS - GeorgiaTech Lorraine
cedric.pradalier@georgiatech-metz.fr

Abstract

This article presents a keyframe-based, innovative map registration scheme for applications that benefit from recurring data acquisition, such as long-term natural environment monitoring. The proposed method consists of a multi-stage pipeline, in which semantic knowledge of the scene is acquired using a pretrained neural network. The semantic knowledge is subsequently employed to constrain the Iterative Closest Point algorithm (ICP). In this article, semantic-aware ICP is used to build keyframes as well as to align them both spatially and temporally, with neighboring keyframes and those captured around the same area but at a different point in time, respectively. Hierarchical clustering of ICP-generated transformations is then used to both eliminate outliers and find alignment consensus, followed by an optimization scheme based on a factor graph that includes loop closure. To evaluate the proposed framework, data was captured using a portable robotic sensor suite consisting of three cameras, a three dimensional lidar, and an inertial navigation system. The data is acquired monthly over 12 months, by revisiting the same trajectory between August 2020 and July 2021.

1 Introduction

The ability to create three-dimensional representations of the natural environment enables scientists to achieve an improved understanding of both natural and man-made changes to nature, such as alterations caused by pollution and human encroachment, often happening at the expense of natural environments.

For instance, slowly occurring changes to the climate affects tree growth, flowering initiation, and the overall natural regeneration cycle. The latter implies the need for long-term monitoring of changes in the natural environment, as well as the ability to align these representations in space and time, in order to perceive natural changes.

*dream.georgiatech-metz.fr

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

Figure 1: The portable robotic sensor suite is a versatile piece of hardware, mounted on a backpack frame. The sensor suite can be carried by a human or affixed to a wheeled robot.

Currently, biologists use manual techniques in order to perceive slowly occurring changes in nature, such as recurrent image acquisition using a handheld camera. However, the latter technique provides little insight on global changes, as the field of view of is limited and suffers, like all cameras, from fluctuating brightness conditions and the challenging task of finding the same scene, at different time periods.

Mapping in natural environment is a challenging task (Chahine and Pradalier, 2018) for both lidars and cameras. Brightness conditions, wet surfaces, semi-transparent surfaces such as tree canopy and the lack of structure are examples of the challenges facing unconstrained mapping algorithms in the natural environment.

Additionally, the state-of-the-art in the field of mapping does not sufficiently explore the concept of mapping over different surveys, henceforth referred to as temporal alignment. The latter implies the alignment of maps taken at different points in time. More specifically and to the best of our knowledge, no previous work explores the task of temporal alignment of natural environment surveys. Nevertheless, the association of natural scenes from different surveys is an important concept for environmentalists looking to qualify and quantify slowly occurring natural changes. The temporal alignment of scenes is also relevant for routine applications involving robot navigation, such as recurrent autonomous inspections of ship hulls and aerofoils.

Appearance-based maps depend on the ability to extract features, yet visual features cannot be reliably and repetitively extracted from natural scenes (Chahine and Pradalier, 2018). Additionally, establishing survey-correspondence and ensuring scene overlap between different takes can be challenging for non-omnidirectional cameras: even with accurate positioning, scene overlap is not guaranteed, whereas omnidirectional cameras suffer from important lens distortion.

Further, fluctuating lighting conditions, snowing surveys and other factors are known to be detrimental to methods that rely on visual inputs, even if semantic knowledge is available. In the experiments, we temporally align a snowy survey with other non-snowy surveys, without any special consideration or post processing. This is despite the fact that the presence of snow has substantially modified semantic ground segmentation, as can be seen in the first 15 seconds of the video in Figure 15.

Natural environment monitoring is an example of an application that can benefit from the temporal alignment of natural environment surveys. For instance, such an approach can replace the need for recurrent manual image acquisition. By aligning natural environment surveys, we show that it is possible to provide both visual and analytical feedback of the changes occurring in the surveyed environment.

Natural changes are slow and sometimes localized, depending on the flora diversity. For instance, one of

many tree species might be affected by evolving conditions such as pollution, whereas other species are not necessarily affected or might not be of interest. Therefore, the perception of natural changes is predominantly localized in character (Rosenzweig et al., 2007). The latter has been confirmed by the many discussions with environmentalists in the Rhine-Meuse area in France. The suggested framework has therefore been adapted to emphasize the detection of local changes around an area of interest. To that end, a keyframe-based approach was adopted.

Keyframes are time-continuous, locally reconstructed maps, often used in the fields of mapping and animation (Younes et al., 2017). In large scale applications such as outdoor mapping, using keyframes facilitates data handling and manipulation. Further, a global optimization scheme can be used to ensure spatial continuity in-between keyframes.

Keyframe-based mapping is the result of either visual or lidar-based mapping techniques. Except for static scanners, maps created using moving platforms, such as a wheeled robot, require the 6 Degrees of Freedom (DoF) pose of the platform in order to properly integrate the map. The state-of-the-art in Simultaneous Localization And Mapping (SLAM) provides tools that can be used to simultaneously build the map while estimating the pose of a moving platform. An example of these algorithms is Direct Sparse Odometry (DSO) (Engel et al., 2017), a sparse visual odometry framework for state estimation and mapping using a monocular camera.

Laser scanners, or lidars, are of the most used sensors when it comes to generating three dimensional representations of the environment. Technological advancement and market accessibility has made lidars more accessible, reinforced by the emergence of solid state variants, that feature low device weight and lower power consumption. The increasing interest in autonomous cars has also driven the need for lidars with new and innovative field of views. For instance, the experiments in this article were conducted using the RoboSense BPearl, a 16-line lidar with a hemispherical field of view, making it suitable for installation on a vertical surface. Such is the case of the portable sensor suite shown in Figure 1.

Lidars are capable of continuously scanning the environment in order to generate high precision point clouds. By aligning consecutive, overlapping point clouds, it is possible to iteratively build a map, by searching for a transformation that aligns two point clouds. Some of the algorithms that align point clouds include the Normal Distribution Transform (NDT) (Biber and Strasser, 2003), and the Iterative Closest Point (ICP) (Besl and Mckay, 1992).

In a previous work (Chahine and Pradalier, 2019), we attempted to use DSO in order to reconstruct the natural environment by stabilizing visual odometry with a basic 2D lidar. However the application was limited due to laser scan sparsity, and the dependence of the method on the overlap between the field of view of a single camera with that of a 2D lidar. More recently, we presented a generalized approach (Chahine et al., 2021) to mapping in natural environment using a factor graph architecture, that can handle multiple cameras, a 3D lidar and an Inertial Navigation System (INS) using a combination of visual SLAM and ICP techniques. The method was found to be robust yet unsuitable for localized map alignment, due to its dependence on a multi-stage global optimization for monocular scale estimation.

The state-of-the-art in mapping and localization therefore exhibits poor performance in natural environment surveys. To the best of our knowledge, no previous work has addressed the three-dimensional temporal alignment of natural environment surveys. The advancement of mapping and localization techniques has however resulted in the formation of a rich toolbox, that contains useful tools such as factor graphs, visual SLAM and ICP. In this paper, the latter techniques are further adapted to challenging environments, such as natural ones, with the purpose of temporally aligning any number of surveys.

We also address the mapping problem by constraining ICP with semantic information (Section 3.2), in order to generate overlapping keyframes. The proposed method assumes that multiple surveys of the same scene are performed, and that the path taken when recording each survey is roughly the same. We then use semantic-aware ICP in the context of a matching scheme (Section 3.3) that lim-

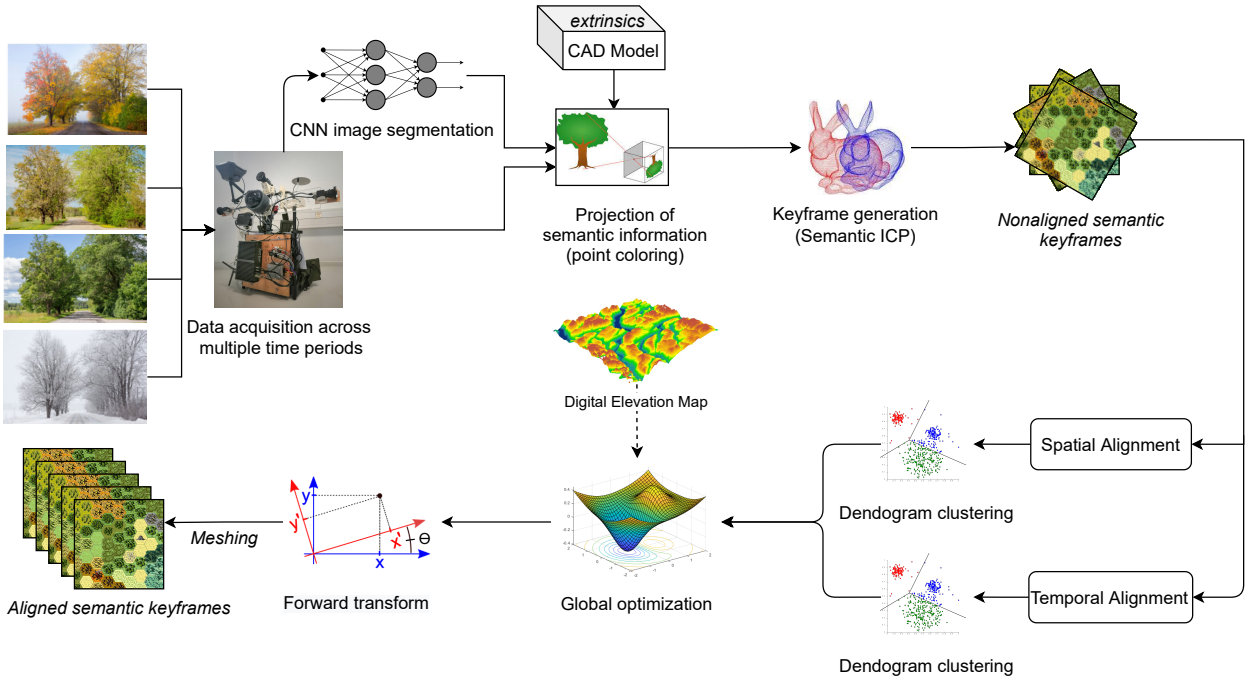


Figure 2: Flow chart of the proposed framework, with dashed arrows representing optional components. Keyframe generation, spatial alignment, temporal alignment and dendogram hierarchical clustering are further detailed in the diagrams shown in Figure 5, Figure 7, Figure 6 and Figure 10, respectively.

its computational complexity, to generate transforms that link keyframes to each other. These transforms are then aggregated, to generate multiple alignment candidates. What follows is the alignment of keyframes from different surveys, in order to create a spatio-temporal representation of the environment. To do so, we exploit the many dimensions afforded by working with multiple surveys, in order to search for a matching consensus. Finally, the proposed framework includes a global optimization scheme, constrained by a loop closure scheme and a Digital Elevation Map (DEM) (Section 3.4). Our contributions are as follows:

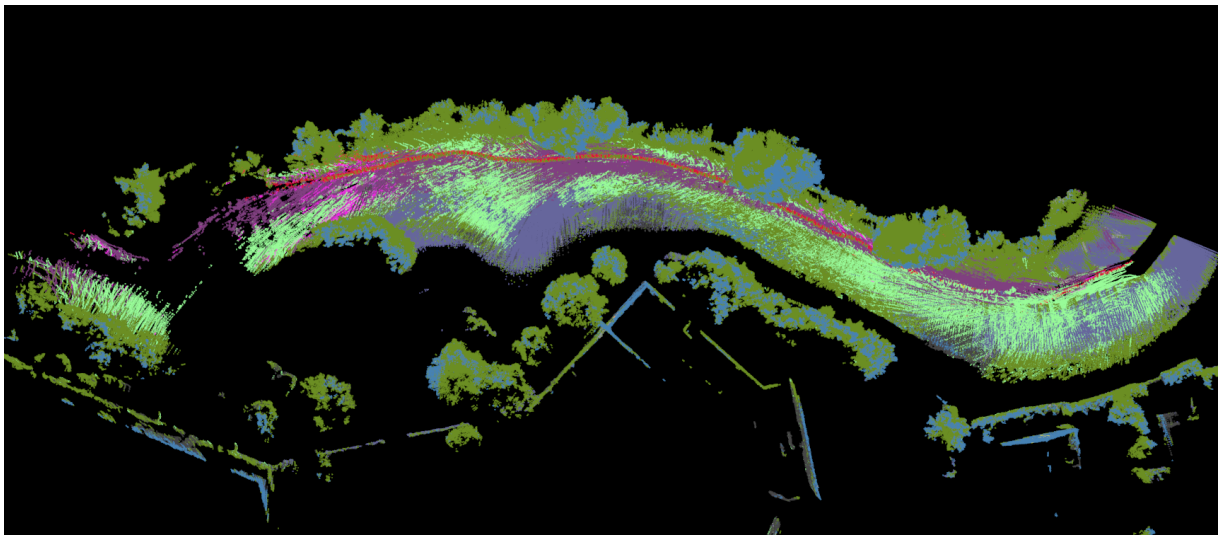
- A mapping framework that takes advantage of semantic-aware ICP (Figure 5) [1].
- A matching scheme for keyframes over multiple surveys (Figure 7 and Figure 6). [2]
- A consensus finding scheme based on hierarchical clustering, followed by an optimization step with loop closure (Figure 10). [3]

2 Related Work

The state-of-the-art does not frequently address the challenges associated with mapping in the natural environment (Chahine and Pradalier, 2018). Nevertheless, specialized applications have driven the need for the development of techniques that can be used to detect changes (Babin et al., 2021), by aligning spatial representations across surveys.

[1] dream-multi-mapper (github.com/georges-chahine/dream-multi-mapper).
[2] dream-multi-matcher (github.com/georges-chahine/dream-multi-matcher).
[3] dream-multi-checker (github.com/georges-chahine/dream-multi-checker).

(a) Google Earth screenshot, 2009.



(b) From the survey of September 2020.

Figure 3: Side-by-side comparison showing a keyframe from the survey of September 2020 and the corresponding snapshot from Google Earth.

Other contributions such as (Biber and Duckett, 2005) present interesting solutions for long-term map building. In the latter work, a temporal laser map registration scheme is presented, that uses wheel odometry and laser scans as input for indoor map registrations, with a semantic layer based on a temporal probabilistic model. Compared to the proposed framework, the work of (Biber and Duckett, 2005) is limited to controlled indoor mapping, whereas wheeled odometry is not available in the unstructured natural environment. Additionally, the slow changes observed in outdoor natural enivronments are fundamentally different from the fast or binary human-induced changes found in an office environment. Moreover, we show that our method is scalable over 12 surveys x 1.3 KM each, captured over a year, whereas the latter work experiments with a 5-week dataset captured in an office hallway.

Some of the work found in literature uses semantic labelling to detect changes. An example of which is the work of (Varghese et al., 2019), where the authors address the problem of detecting public space encroachment from images, using deep learning techniques to label the change. The latter method is however limited to two dimensional images and a particular application. Contrary to (Varghese et al., 2019), the proposed framework aims at providing a rich, semantic representation, that can be manipulated according to the needs of the task at-hand, as further demonstrated in Section 4.5. Further, the proposed method was not designed around the constraints of any particular application.

A couple of recent contributions such as (Guo et al., 2018) and (Sakurada et al., 2020) specifically address the appearance change problem, with the latter paper using semantics to do so. However, detecting changes in image pairs is limited in scope, given the availability of alternative three-dimensional mapping tools. On the other side, our method copies the semantic knowledge from 2D images, to label 3D point clouds, resulting an improved representation of the environment.

The work of (Stent et al., 2015) specializes in temporal alignment for change detection in tunnel surfaces. However, (Stent et al., 2015) uses scale-lacking sparse point clouds, whereas the proposed method uses dense, semantic point clouds, for an improved representation of the environment.

More recently, the state-of-the-art includes methods focusing on the comparison of three-dimensional point cloud pairs, taken at different times (Yew and Lee, 2021). In the latter paper the authors infer, using a neural network, a non-rigid transform to temporally align and account for point cloud inconsistencies. The method is however limited to point cloud pairs, and therefore it is not suited for long-term monitoring over multiple surveys. Conversely, the proposed method is designed to deal with multiple input point clouds.

(Schachtschneider et al., 2017) aims at aligning point clouds captured from self driving cars, using inputs from different surveys in order to detect temporal map changes. The advantage of the latter framework is that it supports multiple inputs, and is able to detect scene changes in a dynamic environment. (Schachtschneider et al., 2017) does not however address the challenges of working in a natural environment such as a forest, and instead avoids to address the mapping problem, by using the Riegl vmx®-250 to solve the localization problem. The use of the vmx®-250 is nevertheless limited to the availability of a Global Position System (GPS) signal, a luxury that cannot be always afforded in a dense forest, and is specifically designed for urban mapping. The proposed method solves both the mapping and the temporal alignment problem, using semantic-aware ICP.

The majority of the above methods are dependent on accurate GPS readings. However, the framework we propose has a very weak dependence on GPS position estimates, a luxury that cannot be afforded while working inside dense forests. This is due to the fact is that only a single GPS reading is required to build and establish keyframe correspondence. Further, GPS repeatability errors can be safely neglected, given the shear size of the keyframes being built.

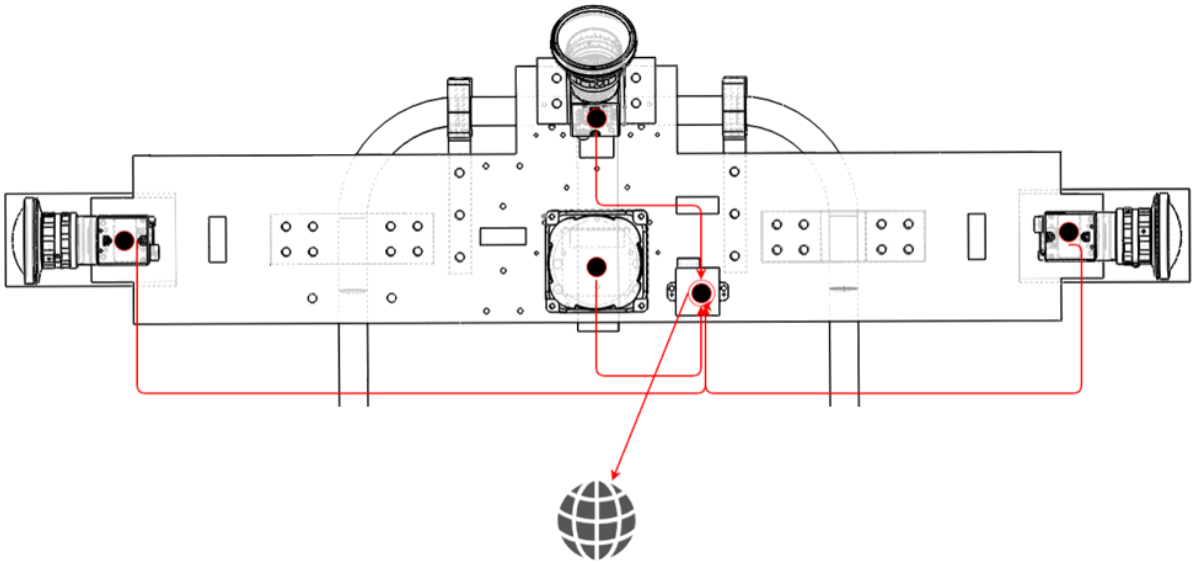


Figure 4: Computer Aided Design (CAD) model of the data acquisition system, used to estimate the extrinsics between cameras, the lidar and the INS. The resulting maps are subsequently referenced in the Universal Transverse Mercator (UTM) system.

3 Framework

In this section, we detail the steps taken to perform temporal or spatio-temporal alignment of keyframes and surveys, respectively. The first step is to semantically segment the images, then copy the semantic knowledge onto 3D points. We then use the acquired semantic knowledge to build keyframes using ICP (3.2). Keyframes are then matched (3.3), and the generated matches are used to find alignment consensus, as well as to generate alignment candidates (3.4). The proposed scheme is further constrained by closing the loop and using a digital elevation map, in a factor graph architecture. Finally, aligned keyframes are filtered, using statistical, semantic and voxel filters, before they undergo meshing (3.6).

In our context, a survey is composed of a series of overlapping keyframes, taken at a given survey. The goal is to align keyframes across different surveys *i.e.*, from different surveys. Additionally, aligning whole surveys implies the stitching of overlapping keyframes both in space and time.

3.1 Semantic Knowledge

When surveying natural environments, classical mapping techniques such as ICP often require additional constraints to properly converge (Chahine et al., 2021). To do so, we propose to constrain ICP using semantic information, extracted from the pixel segmentation of images.

This work therefore uses a pretrained neural network for semantic image segmentation (Larsson et al., 2019). The latter model has been trained by its authors using the cityscape dataset (Cordts et al., 2016). The neural network can identify 19 classes of objects such as vegetation, terrain, ground, sky, persons and buildings.

Using this network, we obtain a semantic segmentation of acquired images. In our experimental setup, we segment all the images from all three cameras shown in Figure 1. The semantic knowledge is then copied to the laser points, by projecting the laser rays onto the camera image planes.

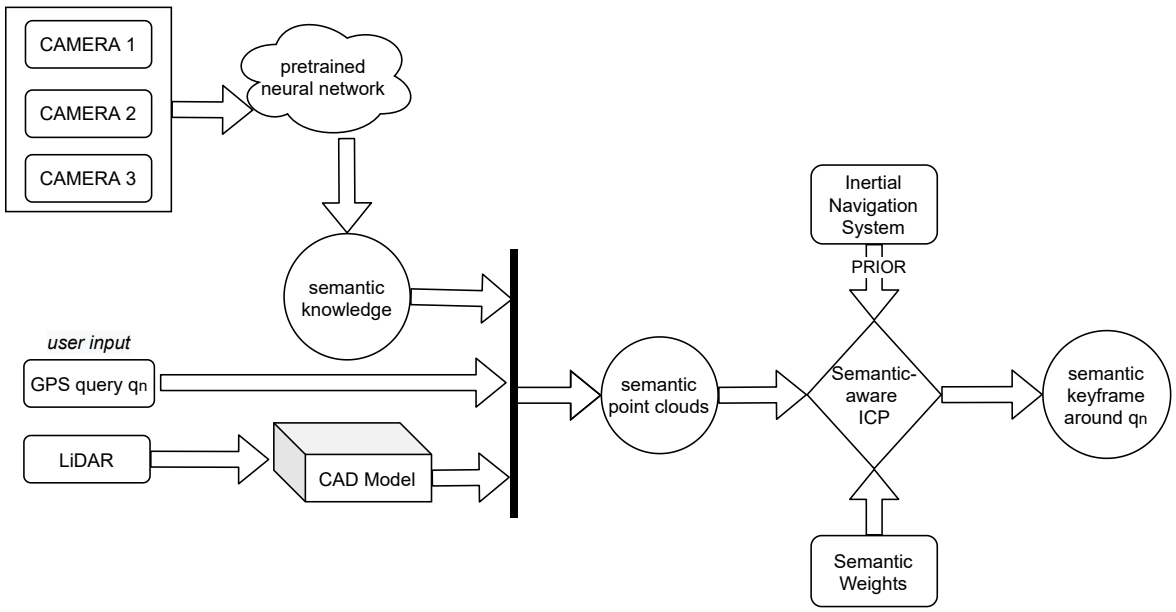


Figure 5: Regular point clouds generated by a LiDAR are converted to semantic point clouds, by copying the semantic knowledge from images to laser points, using the sensor suite's CAD model. Semantic-aware ICP is then used to solve the SLAM problem, using the INS 6-DoF increment as a prior to the state.

3.2 Semantic-aware ICP and keyframe generation

In this section, keyframes to-be-matched are generated using a constrained version of ICP. Further, it is convenient to construct keyframes around continuous query points, obtained by discretizing the GPS trajectory. This has the advantage of facilitating keyframe correspondence in Section 3.3.

This paper proposes a modified, semantic-aware ICP, for both keyframe reconstruction and the temporal alignment of keyframes. ICP is subsequently constrained:

- On the ICP matcher level:
  - By rejecting point-pairs that belong to different semantic classes.
  - By assigning custom weights to point-pairs that belong to similar semantic classes.
- On the data input level: by filtering points that belong to certain semantic classes, such as dynamic objects (pedestrians) that could adversely affect the alignment quality.

Keyframes are then iteratively reconstructed using semantic-aware ICP, around a 2D GPS query point, as shown in Figure 5. Still in the same figure, semantic weights affect the weights given to match pairs in the ICP matcher, and are also used to filter out dynamic semantic classes such as pedestrians. Further, at the input level, a semantic descriptor filter deletes points with a weight that equals to zero.

As for the scan matching process, the proposed method uses a scan-to-map approach, where every new lidar scan is matched to all previously matched scans in a keyframe. What follows is the forward transformation of the pose into the world frame:

$$M = T_0 T_l^b T_i \tag{1}$$

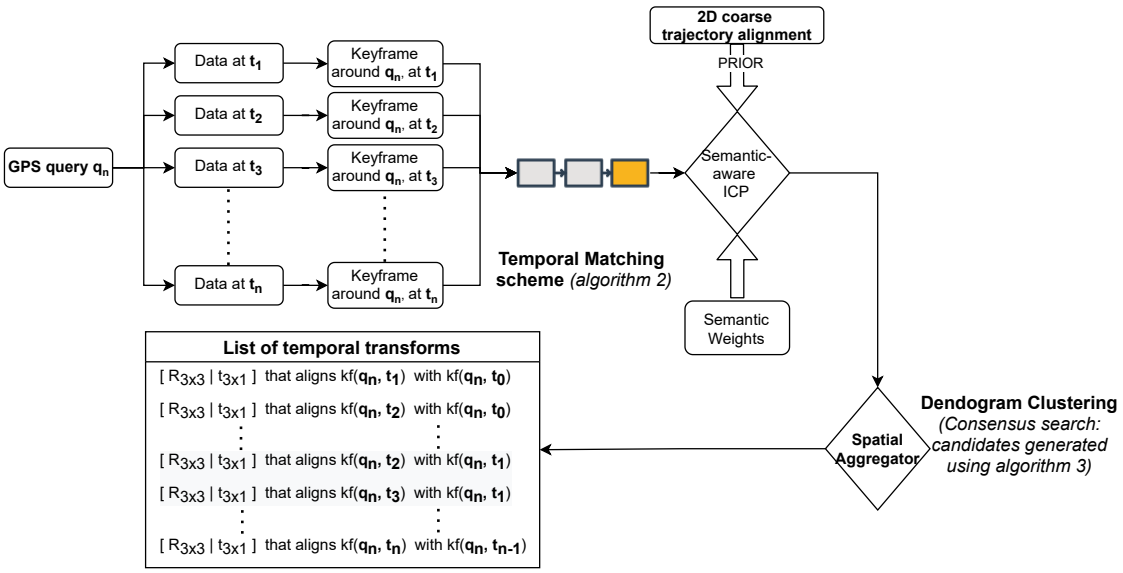


Figure 6: Temporal alignment: keyframes constructed around the same query are aligned using a temporal matching scheme, further detailed in algorithm 2. Subsequently, temporal matching candidates are generated using semantic-aware ICP, with a two-dimensional prior inferred by aligning keyframe trajectories. Finally, matching candidates are re-arranged according to algorithm 3 and fed into an aggregation and consensus finding scheme, further detailed in Figure 10.

T_i is the ICP scan-to-map transform, T_l^b is the extrinsic matrix describing the transformation from the lidar to the common robot frame, also known as base link. T_0 is the full 6 DoF pose of the sensor suite at $t = 0$ *i.e.*, at the start of the keyframe.

The proposed method assumes that the path taken when recording each survey is roughly the same. When automatically generating keyframes for a list of surveys, the trajectory is discretized into several query points while processing the first survey. The same discretization *i.e.*, the same query points are subsequently used to generate the keyframes for all other surveys. This allows the simplification of the temporal matching process, for instance: keyframe 0 in survey 0 is matched with keyframe 0 in surveys 1, 2, 3, ...n. For convenience, query points are communicated in the global latitude/longitude system, with a followup internal conversion to UTM for map registration. For the remainder of this article, keyframe α at survey β is represented as $S(\alpha, \beta)$.

3.2.1 Overlapping keyframes

At a later stage (Section 3.3), the spatial alignment of keyframes is performed using ICP-generated transforms. However, since ICP requires an overlap between its inputs, then a continuous sequence of overlapping keyframes is generated for every survey. In the experimental section, keyframe overlap is enforced by choosing a keyframe radius that is significantly larger than the distance in-between queries. Nevertheless, the method in which the overlap is enforced is not critical to the proposed alignment scheme. Alternative methods to create overlapping keyframes is to append a non-overlapping keyframe to the previous one.

The iterative reconstruction of a keyframe is done, in part, by reusing previously found ICP increments, from previous overlapping keyframes. Such an approach has the advantage of minimizing spatial alignment errors, given that point clouds are randomly downsampled in the matching process. As a result, ICP is known not to generate the exact same result even with the same inputs. The reconstruction of an overlapping keyframe is therefore a two-stage process. The first stage consists of identifying, using lidar scan timestamps, previous ICP runs and calculating the ICP increment, followed by a forward transformation into the frame of reference

of the new keyframe. The reconstruction of non-overlapping parts uses the last found transformation as a prior to the now resumed ICP.

Algorithm 1: Reconstruction of an overlapping keyframe

Result: Overlapping keyframe
increments=[] ; // empty list
timestamps=[] ; // empty list
map=[] ; // empty list
P=$I_{4\times4}$; // ICP prior initialized to identity
for *every scan* sc_i *in a keyframe* $S(z,i)$ *at time* t_i **do**
  $T_i = I_{4\times4}$;
  if $t_i > timestamps(end)$ **then**
    $T_i = icp(sc_i \in S(z,i), P)$;
    $T_{incr} = T_{i-1}^{-1} T_i$;
    append$(increments, T_{incr})$;
    append$(timestamps, t_i)$;
  else
    find l such as $timestamps[l] = t_i$;
    $T_{incr} = increments[l]$;
    $T_i = T_i T_{incr}$;
  end
  $P = T_i$;
  $S_i = T_0 T_l^b T_i sc_i$; // Equation 1
  append(map, Si);
end

In our implementation of algorithm 1, the ICP prior has been forward transformed by the INS increment between the current timestamp and the previous one: $P_i = P_{i-1} T_{ins_{i-1}} T_{ins_i}$

3.3 Spatio-temporal alignment

In this section, we detail the steps taken to generate multiple keyframe match estimates using semantic-aware ICP. We also adopt a matching scheme, given the high dimensional aspect of the problem.

The proposed method is designed to handle multiple surveys taken at different points in times. A keyframe matching scheme is subsequently implemented, where every keyframe is matched, using semantic-aware ICP to the previous keyframe in space (Figure 7), and to all other keyframes in time (Figure 6), in the latter case constructed around the same query.

Let S_i be a survey in a list of m surveys such as $i < m$. For every survey S_i, let $S(z,i)$ be a keyframe in a list of n keyframes. As previously mentioned in Section 3.2, keyframe $S(0,0)$ in S_0 would be the natural candidate to match with keyframes $S(0,\forall)$ in $S_1, S_2, ...S_m$, as they are all constructed around the same query point.

The ICP prior for spatial matches is calculated by matching timestamps in-between overlapping keyframes from the same survey, since the first pose in a keyframe has been observed in the previous (overlapping) keyframe. As for temporal matches, a coarse, brute force alignment of the keyframe-associated 2D trajectories is performed to obtain the prior to ICP.

To calculate the temporal alignment prior, trajectory pairs are aligned in 2D, first by aligning trajectory medians, followed by a minimization of the squared distance between trajectory points. To find the alignment prior, the brute search process uses one meter increments along the horizontal and vertical directions for

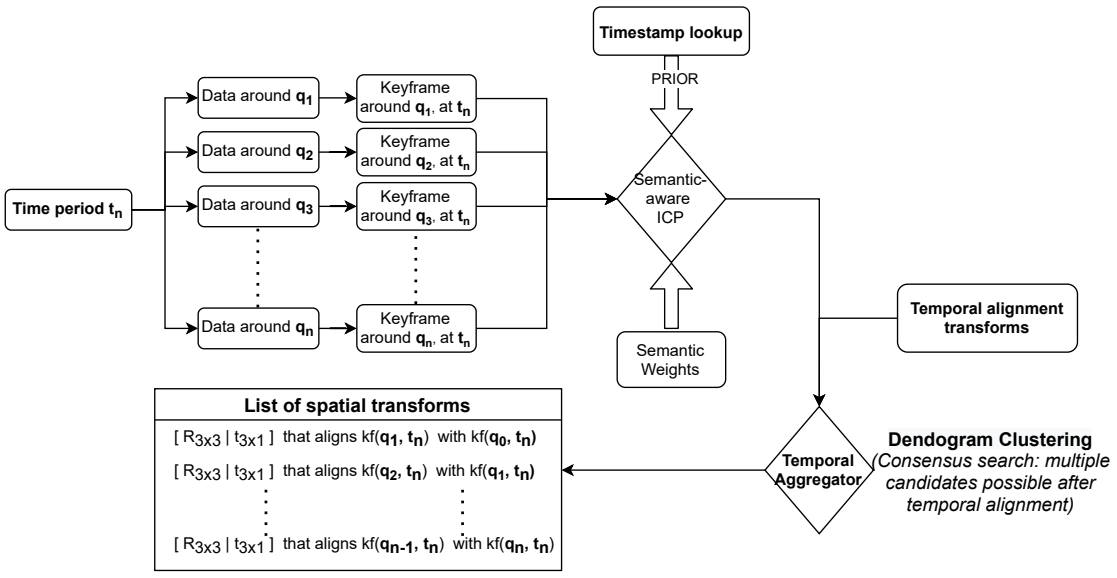


Figure 7: Spatial alignment of overlapping keyframes belonging to the same survey. The Spatial alignment is a by-product of temporally aligned overlapping keyframes. Overlapping pairs are matched using ICP with a prior inferred using timestamp lookup on the associated trajectories *i.e.*, by calculating the increment between the first pose in a keyframe and a corresponding pose in the overlapping keyframe, found by matching timestamps.

translation, and one degree increments for the rotation part.

What remains is to design a matching scheme *i.e.*, to select keyframe match pairs for spatial and temporal ICP matches. The spatial matching scheme consists of matching every keyframe to the previous overlapping keyframe, whereas the temporal matching scheme consists of matching a given keyframe to all other surveys, without replacement. The temporal matching scheme is shown in algorithm 2, where X_z is the trajectory associated with a keyframe $S(z, i)$.

Algorithm 2: Temporal matching scheme

Result: List of temporal matches
logger=[]; ; `// empty list`
for *every survey* S_i **do**
  for *every other survey* S_j *such as* $j > i$ **do**
    for *every pair of keyframes* $S(z,(i,j))$ *in surveys* S_i *and* S_j **do**
      P=$get\_prior(X_z \in S_i, X_z \in S_j)$;
      transform=icp$(S(z,i), S(z,j), P)$;
      append(logger,transform);
    end
  end
end

The total number of keyframe matches implied by algorithm 2 is:

$$n \times m(m-1)/2 \tag{2}$$

which roughly translates into 4620 candidate matches or ICP runs for $m = 12$ surveys, each containing $n = 70$ keyframes.

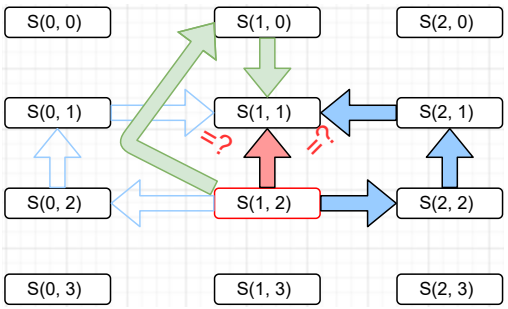


Figure 8: A graphical interpretation for the generation of temporal alignment candidates, detailed in algorithm 3. As a reminder, i in $S(i, j)$ represents a keyframe number, and j represents a survey. Keyframes with the same i are queried around the same position but for different surveys, with both i and j being continuous in space and time, respectively.

For the remainder of this article, we would like to clarify that the term "candidate matches" refers to the above calculated ICP transforms. In the upcoming section, the term "alignment candidates" refers to any transform that aligns a keyframe to another, whether it is composed from a single candidate match, or several. The generation of alignment candidates is detailed in Section 3.4.1.

3.4 Search for consensus and optimization

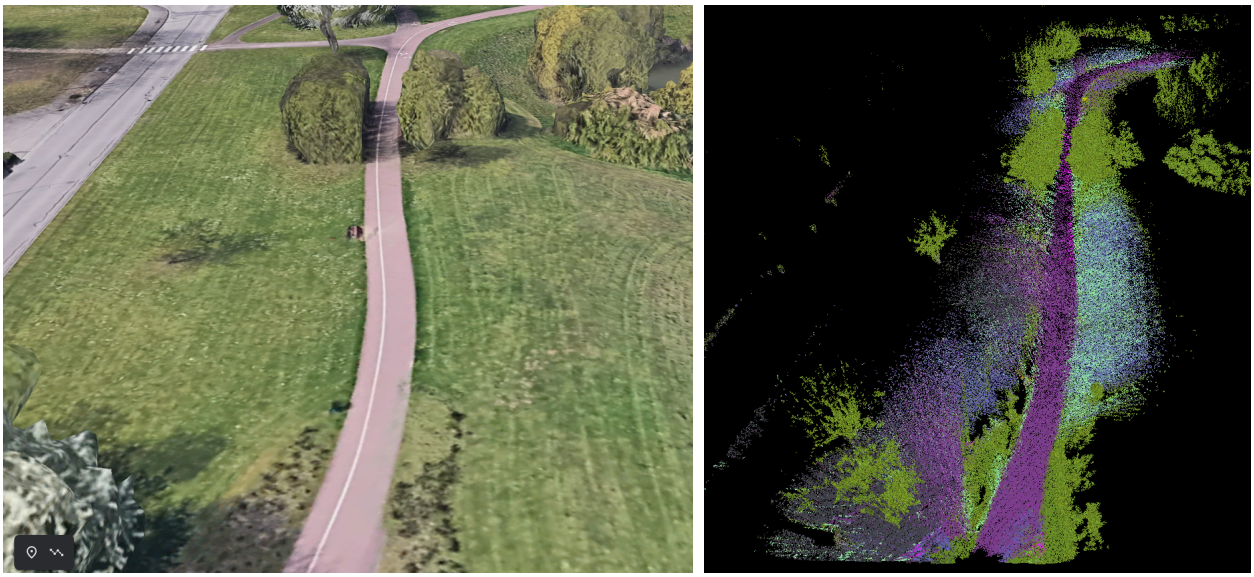
In the previous section, we generated a list of transformations that covers all possible alignment options between keyframes queried at a given survey, and between overlapping keyframes. Here, candidate matches are processed to create kinematic chains that align keyframes. We therefore capitalize on the multi-dimensional aspect of the problem, in order to generate multiple alignment candidates, by searching the previously found list of transformations.

The queried transformations are then combined to form alignment candidates, followed by a search for consensus. The term clustering refers to hierarchical cluster analysis (Cecil C. Bridges, 1966), based on a metric reflects on the measure of dissimilarity between clusters. Since we would like to compare alignment candidates, which are geometric transforms in the $SE(3)$ space, the following metric is defined.

Two or more transformations form a consensus, or a cluster, if the distance in $SE(3)$ between them, consisting of the geodesic for the rotation part, and the translation norm for translation part is constrained below a certain threshold. The chosen data metric is calculated as follows. Let T_1 and T_2 be two transforms that represent two alignment candidates. Let T' be the increment between the two transforms, as in $T' = T_1^{-1}T_2$. Let $\|T_x\|$ be the norm of the translation part, and let $\boldsymbol{\theta}$ be the geodesic angle norm in the axis-angle representation of T', estimated as the logarithm of the rotation part of T', or $log(T_R)$ (Barfoot, 2017). Further, let k_t and k_r be two weights with an initial value of 0.01. We now calculate the data metric used in the aggregation of $SE(3)$ transforms (Equation 3).

$$d(T_1, T_2) = ||T'||_{SE(3)} = \frac{1}{k_t}||T_x|| + \frac{1}{k_r}||log(T_R)|| \tag{3}$$

The above metric is then used as a measure of dissimilarity between observations. The numerical significance of the data metric shown in the above equation lies in the weights used. For instance, if $k_t = k_r = 0.1$ then a 0.1 meter of translation is equally penalizing as 0.1 radians of rotation. The smaller k_t and k_r are chosen, the more closely knit cluster members are. In the proposed implementation, k_t and k_r are incrementally increased in steps of 0.05 until a cluster is found. Additionally, a link (or observation) is discarded if the values of k_t and k_r exceed 0.5 without finding a consensus.



(a) Google earth screenshot, 2009.

(b) From the survey of December 2020.

Figure 9: Side-by-side comparison showing a keyframe from the survey of December 2020 and the corresponding snapshot from Google Earth.

3.4.1 Generation of alignment candidates

Candidate matches are then used generate multiple alignment options for each keyframe. A sample showing at least two alignment options between two keyframes is shown in Figure 8. Still in the same figure, it is possible to align keyframe 1 at time period 2 with keyframe 1 at time period 1, by inferring the transformation directly from candidate matches, marked by a red arrow in the mentioned figure. Alternatively, the alignment can be achieved by going though keyframe 1 at time period 0, marked by green arrows. Furthermore, it is possible to make two jumps before reaching the desired keyframe and time period, an example shown by blue arrows. The proposed method however limits the number of intermediate jumps to two, to prevent error accumulation, and to reduce problem complexity, with the presence of a high number of keyframes and time periods.

Groups to be clustered are therefore formed by aggregating candidate matches over all possible paths. The implications of the data generated by the matching scheme in Section 3.3 further constrains intermediate jumps to either within the same keyframe number *i.e.*, vertical arrows (Figure 8), or within the same survey by jumping to the next, or previous, overlapping keyframe *i.e.*, horizontal arrows. (Figure 8).

The proposed general solution for the multiple path problem is presented as follows:

Let T be a list of candidate transforms that aligns keyframe i at time period j, henceforth known as $S(i, j)$, with $S(i, j-1)$, such that $0 \leq i-1 < i+1 \leq m$ and $0 \leq j-1 < j+1 \leq n$, where m and n are the total number of keyframes and time periods (surveys), respectively. Let $query\_tf(i_1, j_1, i_2, j_2)$ be a query function that performs value lookup in the transformations previously found in Section 3.3. What follows is the generation of alignment candidates for $S(i, j)$ to $S(i, j-1)$, shown in algorithm 3.

Across n time periods, the total number of alignment candidates for $S(i, j)$ with $S(i, j-1)$ is calculated as $3 + (n-2) = n+1$ alignment candidates for each keyframe.

The transformation matrices are then fed into a multi-stage hierarchical clustering algorithm in $SE(3)$.

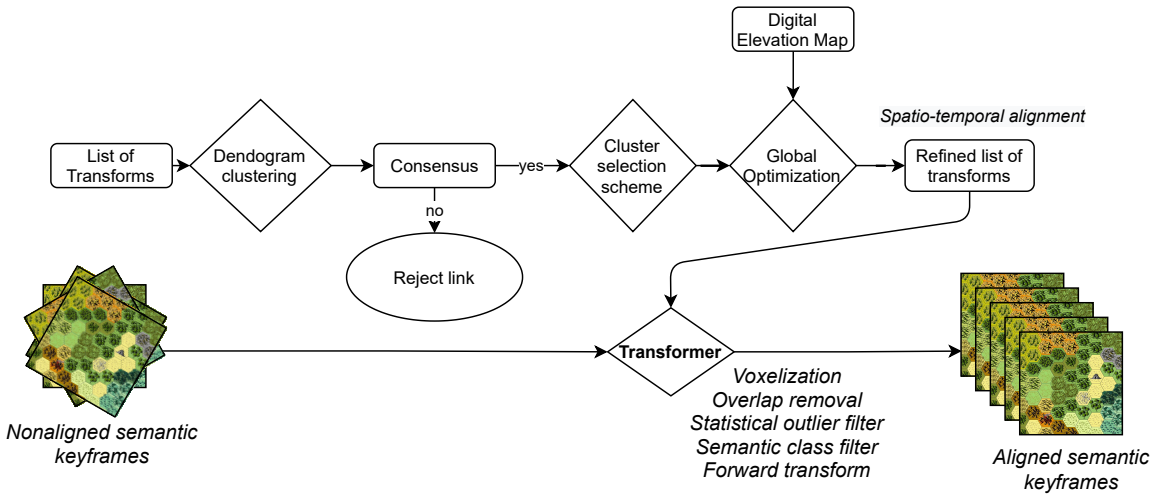


Figure 10: Matching candidates generated by algorithm 3 are then clustered using dendogram clustering. In the majority of cases, multiple clusters are expected. A selection scheme therefore selects the parent keyframe with the most tightly knit clusters. Finally, a graph-based optimization scheme allows the integration of a DEM into the framework and activates the loop closure constraint.

The final selection is made on the biggest cluster or the cluster with the smallest standard deviation, if tied. Rotation and translation components of the chosen cluster elements are then averaged using Spherical Linear Interpolation (SLERP) (Shoemake, 1985) and classic linear interpolation, respectively. If no consensus is to be found below the chosen threshold, then the concerned link is discarded.

Additionally, a common frame of reference for each keyframe across all surveys needs to be chosen. In other words, there is a need for the designation of a parent keyframe that sets the frame, along which all other

corresponding keyframes will align.

Algorithm 3: Generation of alignment candidates between two keyframes

Result: Candidate transforms for aligning keyframes $S(i,j)$ with $S(i,j-1)$.

T=[] ; `// empty list of transforms`
$tf = query\_tf(i,j,i,j-1)$;
append (T, tf);
$tf_1 = query\_tf(i,j,i+1,j)$;
$tf_2 = query\_tf(i+1,j,i+1,j-1)$;
$tf_3 = query\_tf(i+1,j-1,i,j-1)$;
$tf = tf_3 * tf_2 * tf_1$;
append (T, tf);
$tf_1 = query\_tf(i,j,i-1,j)$;
$tf_2 = query\_tf(i-1,j,i-1,j-1)$;
$tf_3 = query\_tf(i-1,j-1,i,j-1)$;
$tf = tf_3 * tf_2 * tf_1$;
append (T, tf);
for *every time period* x **do**
  if $x \neq j$ ***and*** $x \neq j-1$ **then**
    $tf_1 = query\_tf(i,j,i,x)$;
    $tf_2 = query\_tf(i,x,i,j-1)$;
    $tf = tf_2 * tf_1$;
    append (T, tf);
  end
end

The parent frame is selected in order of priority:

1. The frame which aligns with the least number of discarded links
2. If tied, the frame which aligns with the lowest average of an error term, calculated by averaging out an $SE(3)$ distance metric between cluster centers and their respective members, as shown in Equation 4.

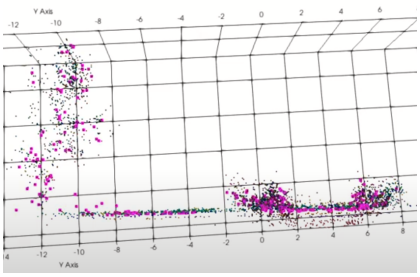
$$E = \|x\| + log(R) \tag{4}$$

Where E is the distance metric between the cluster center and one of its elements, all expressed in $SE(3)$. $\|x\|$ is the norm of the difference between the 2 translation vectors and R is the rotation associated with the rotation increment between the two attitudes, *i.e.*, between the cluster center and the chosen element.

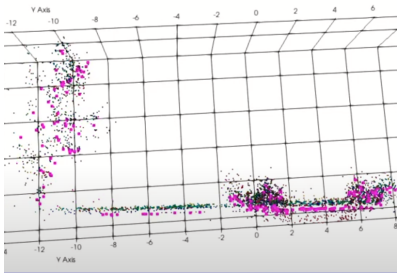
As for spatial alignment, alignment candidates are generated in the frame of reference of the previously found parent frames, followed by the same hierarchical clustering algorithm. Aligning in the frame of reference of the parent frame has the advantage of simultaneously spatially aligning a block of keyframes across all surveys, with its overlapping neighbor block of keyframes, also expressed in the frame of its own parent.

3.5 Loop closure and optimization

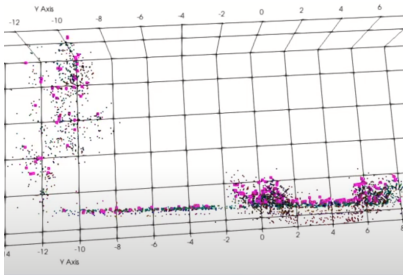
A survey is assumed to be a spatially continuous set of keyframes. As previously mentioned, it is assumed that the same trajectory is roughly followed during each survey, although the starting point for every survey can be different.



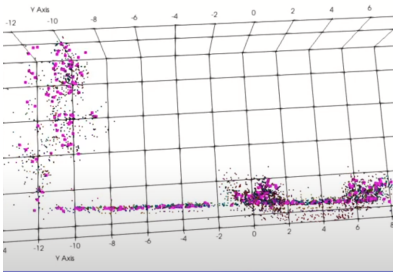
(a) From the survey of August 2020



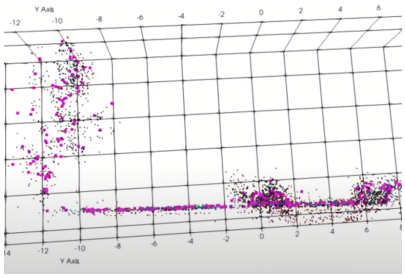
(b) From the survey of September 2020



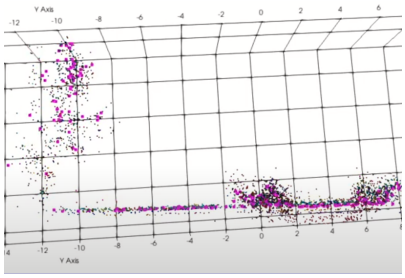
(c) From the survey of October 2020



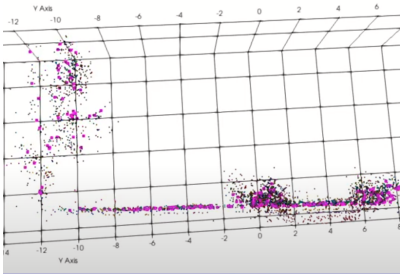
(d) From the survey of November 2020



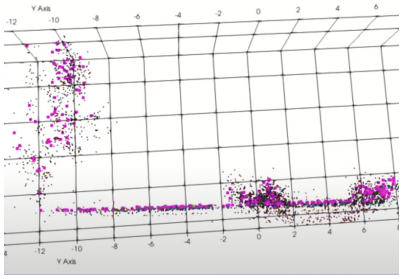
(e) From the survey of December 2020



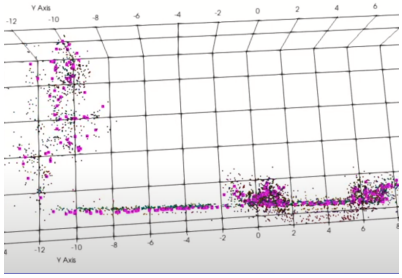
(f) From the survey of January 2021



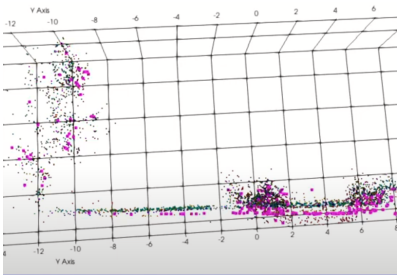
(g) From the survey of February 2021



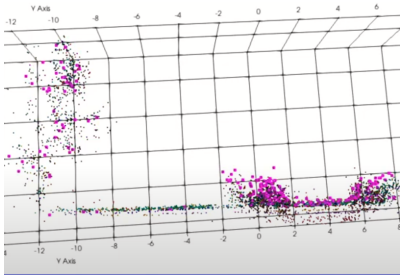
(h) From the survey of March 2021



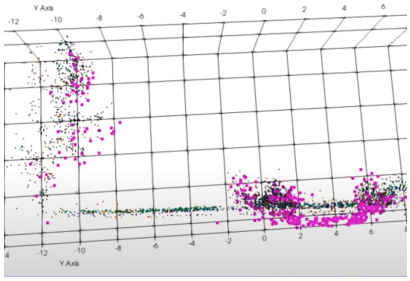
(i) From the survey of April 2021



(j) From the survey of May 2021



(k) From the survey of June 2021



(l) From the survey of July 2021

Figure 11: Cross-sectional view of 12 aligned keyframes, shown on-top of each other against a 2x2 m. grid. Points in pink correspond to the survey shown in the associated figure caption. In the video, a cross-section of the above is performed, to show the distribution of points from different surveys against a grid. The corresponding animation as well as the method used to generate it can be seen here (https://youtu.be/YhE9wEt2kMQ).

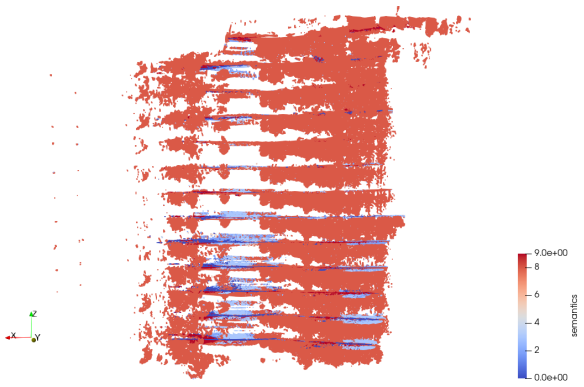
If the trajectory loops, then it is possible to close the loop in order to reduce and correct for error accumulation sources, such as positional drift. Further, it is assumed that the same behavior, *i.e.*, the same movement pattern, is roughly copied in surveys taken at different time periods. We therefore aim to exploit the temporal aspect of the loop closure problem.

Loop closure, is performed by geographic query of revisited parts, using timestamps as well as the trajectories to detect if a keyframe is being revisited, after being initially exited. This particular scheme is further detailed in (Chahine et al., 2021). The keyframes are then matched using ICP. This is repeated over all surveys, building on the assumption of a repeated behavior.

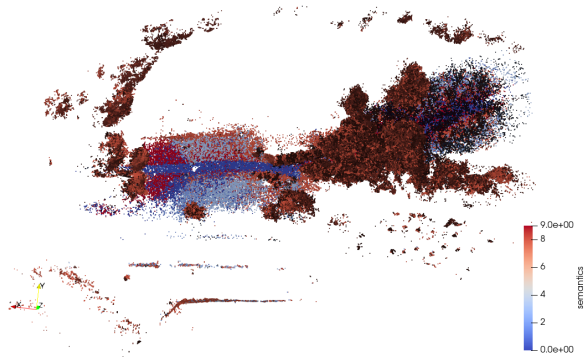
The revisited keyframes for loop closure are then aligned to their respective parent keyframes, which allows the comparison of the $SE(3)$ candidate transforms across all time periods. The latter group is then clustered using the same hierarchical clustering scheme used in Section 3.4.

3.5.1 Factor graph

Factor graphs are flexible tools that can be used to fuse data coming from different sources. A factor graph is composed of nodes, interconnected by constraints such as transformation increments, as well as priors containing absolute knowledge, such as a DEM. An initial estimate is subsequently refined during an optimization and smoothing step.



(a) Exploded view of a keyframe aligned over 12 surveys captured on-foot over 12 months, plus one additional keyframe captured around the same location using the clearpath Husky Unmanned Ground Vehicle (UGV). Animation: here (https://youtu.be/z9EeinFAd78).



(b) Top view of 12 aligned keyframes stacked on top of each other, captured on-foot over 12 months. A video showing the above point clouds being manipulated is available here.

Figure 12: Multi-view and animated point cloud manipulation of the temporal alignment of a keyframe captured around $(latitude, longitude) = (49.103622, 6.216097)$.

Using a factor graph architecture, we propose a non-linear least squares optimization, whose purpose is to constrain and optimize the spatio-temporal kinematic chain, by closing the loop and using a DEM data to sparsely constrain the z-component.

The proposed factor graph is composed of nodes representing serialized keyframe numbers and time periods, for a total of $m \times n$ nodes. These nodes are connected by constraints, composed of the chosen spatio-temporal alignment increments, with respect to the parent frames. To these constraints we add the loop closure cluster average transformation, if a clustering consensus is found. Finally, we also add a z-component prior, also known as a unary factor, to constrain the keyframe altitude according to the DEM.

The factor graph, shown in Figure 13, is subsequently optimized and a refined list of spatio-temporal alignment candidates is obtained. In the experiments, the g2o (Kuemmerle et al., 2011) framework was used,

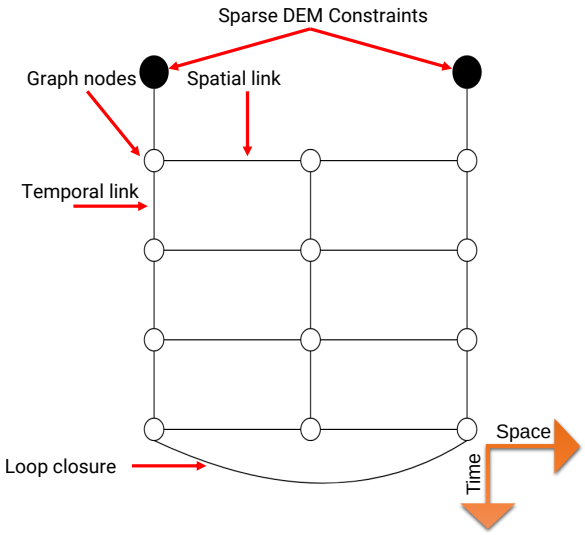


Figure 13: The proposed factor graph architecture, with a loop closure and sparse DEM constraints.

with a robust Cauchy kernel and a classic Levenberg–Marquardt (Moré, 1978) solver for the underlying least-squares problem. Factor graph node priors are the spatio-temporal alignment candidates that belong to the chosen parent frame, for every keyframe. We will also use the clustering metric as a measure of confidence *i.e.*, k_r and k_t for the rotation and translation components, respectively.

3.6 Meshing, filtering and map visualization

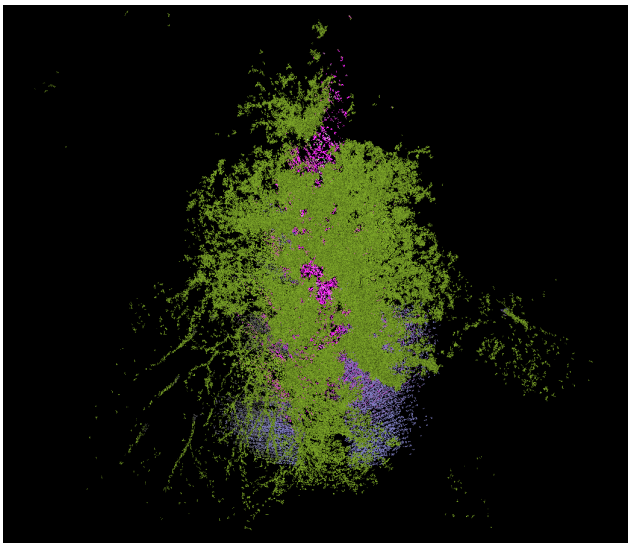
Parameter	Value
Voxel size	0.1
Voxel decomposition type	Tetrahedron
Number of normals for distance function evaluation(k_d)	10
Edge collapse percentage	0.1
Number of normals used in the interpolation process(k_i)	10
Removal of dangling artifacts	100
Number of normals used for normal estimation(k_n)	300
Small region threshold	30
Grid extrusion	False
Surface optimization	False
Maximum size for hole filling	100
Tessellation	False

Table 1: Meshing parameters using the Las Vegas Reconstruction Toolkit (Wiemann et al., 2012)

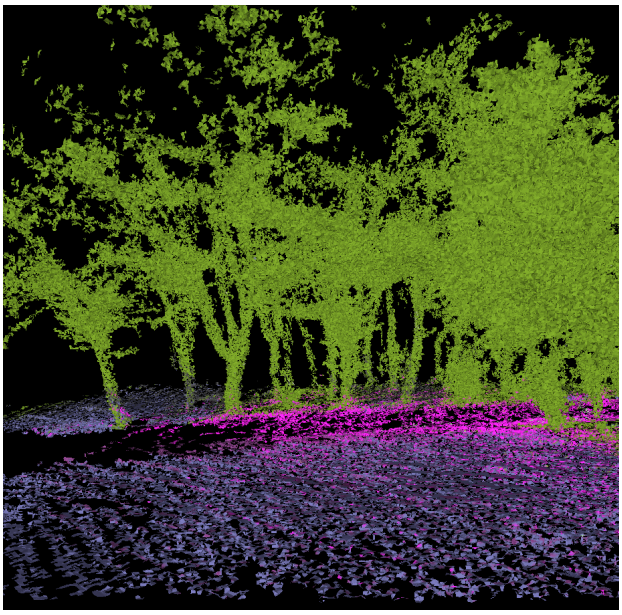
In this section, the now aligned point clouds are further processed, in an effort to produce visually appealing maps that can also be used for change detection. To do so, point clouds are filtered using a statistical filter, followed by a semantic class filter. Point clouds are then voxelized, meshed and then exported for visualization.

After optimization, the keyframes and their associated trajectories are re-exported by applying the previously refined alignment. During the export step, points associated with selected semantic classes are filtered, based on the application at hand. Outlier points, that are more than one standard deviation from their ten nearest neighbors, are also filtered using a statistical outlier filter. The overlapping parts of keyframes are also filtered out, using timestamps, by removing points that have already been observed in the previous keyframe.

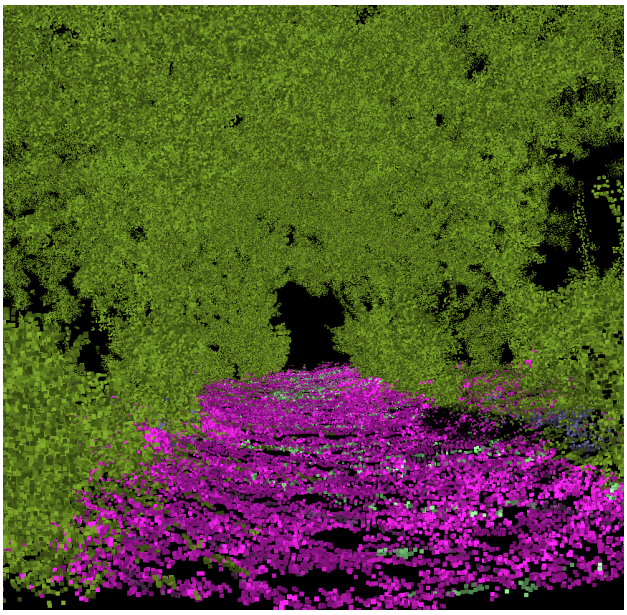
(a) Maximum zoom using Google Earth, 2009.



(b) From the survey of January 2021, top view.



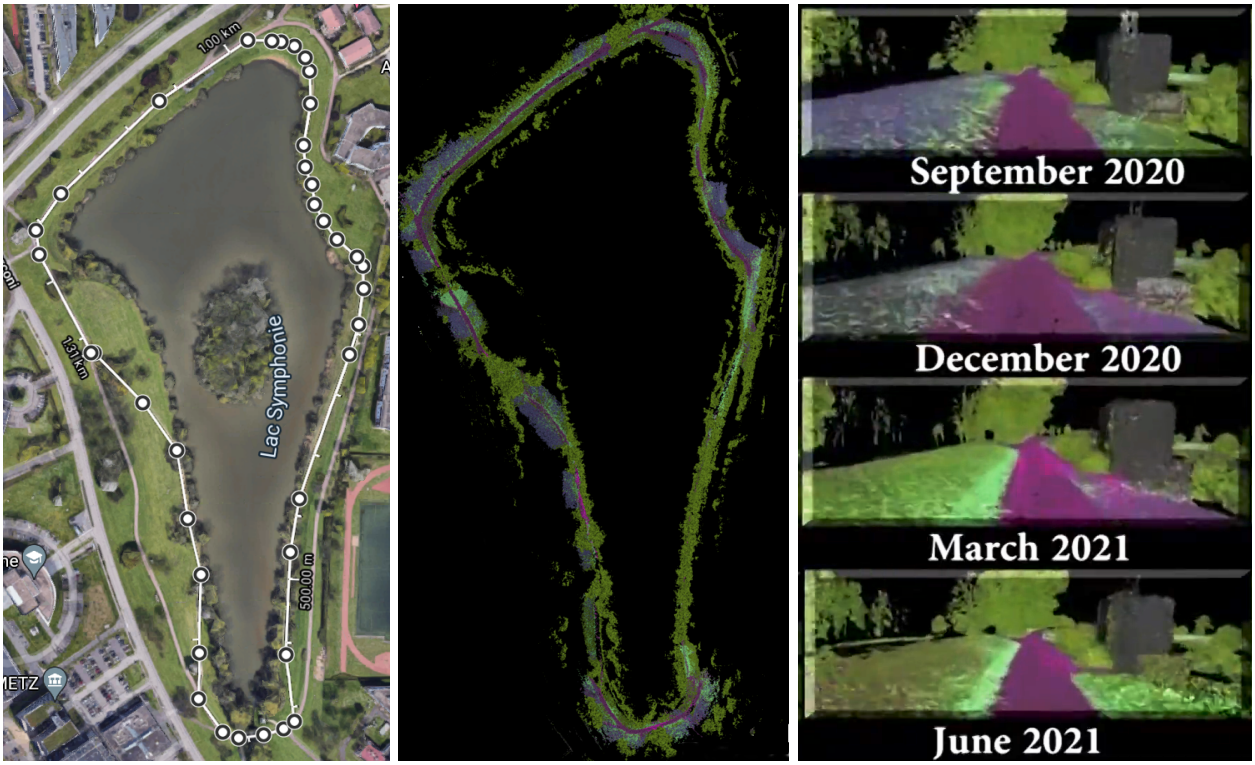
(c) From the survey of January 2021, side view.



(d) From the survey of January 2021, cross-sectional view inside the canopy .

Figure 14: Side-by-side comparison of a keyframe from different view points, taken from the survey of January 2021 and the corresponding snapshot from Google Earth. Note that it is not possible to perceive the inner part of the above canopy, using Google Earth.

Point clouds are then meshed, using the Las Vegas Reconstruction (LVR) toolkit, without grid extrusion, as it was found to significantly increase the number of artificially induced artifacts. The latter is due to the absence of planar surfaces, and due to the chaotic nature of an unstructured natural environment. The meshed point clouds are then visualized using VTK (Schroeder et al., 2006) and Paraview (Ahrens et al., 2005), for an in-depth navigation of the meshed point clouds. An example of such navigation, captured on a small boat in the Orne river is shown here (https://youtu.be/K709A0ckggI).



(a) Snapshot from Google Maps showing the 1.31 KM trajectory taken during each survey. White control points are required to draw the trajectory on Maps.

(b) Corresponding reconstruction, here using the survey of March 2021

(c) Snapshot of an in-depth navigation of aligned and meshed point clouds across multiple surveys. The full video can is referenced in the below caption.

Figure 15: Three-dimension reconstruction of the area around lake Symphony in Metz, France. A more detailed insight can be acquired by watching the following video in 1080p: click here (https://youtu.be/iB0Qn1Xeuog).

4 Results and Discussion

4.1 About the data

The dataset was captured using the portable robotic sensor suite shown in Figure 1. The data acquisition device consists of two cameras looking sideways and one camera looking backwards. All cameras are equipped with a 3.5mm lens, with a field of view of 117° along the the diagonal. Additionally, the RS-bPearl, a 16-line lidar with a hemispherical field of view is facing backwards as well, with a significant overlap with the field of view of all 3 cameras. Moreover, an INS provides a noisy estimate of the 6-DoF pose by fusing GPS and Inertial Measurement Unit (IMU) estimates. For an improved estimate of the extrinsics, all sensor fixtures were 3D printed using design shown in Figure 4. Further, the sensors are externally pulse-synchronized.

The data was captured over twelve months spanning between August 3, 2020 and July 1, 2021, around lake Symphony in Metz, France. The trajectory taken, drawn in Figure 15a consists mostly of mud trails with minimal human intervention, in places spaced with bike lanes, surrounded by trees and the lake itself. The surveys of January and February 2021 were recorded in exceptionally snowy conditions.

4.2 Generation of semantic keyframes

The data, recorded using ROS (Stanford Artificial Intelligence Laboratory et al., 2018), was processed after each data acquisition. Post-processing consisted of extracting the images, followed by the semantic segmentation of the latter images using a pre-trained neural network (Larsson et al., 2019) from literature. The uncompressed size of a survey is around 150 GB, for a total of around 1.7 TB of data. What follows is the transfer of semantic knowledge from the classified images to the lidar point cloud. An example of such classification can be seen here (https://youtu.be/dYBxGqGkqmY). For practical reasons, and due to the sheer size of the data, the compressible rosBAG format was found to be well suited in keeping the data compact and easily accessible, even after introducing systemic changes.

Afterwards, we administer semantics onto lidar points, a process during which semantic information is re-encoded into lidar data. The pinhole model was used to transfer the semantic knowledge from each camera's frame of reference, onto the lidar's frame of reference: $p = A[R|t]P_w$, where P_w is the 3D point to-be-colored, $[R|t]$ is the extrinsic matrix that provides the geometric connection between lidar and camera frames, A is the camera instrinc matrix, obtained by checkerboard calibration and $p(u, v, 1)$ is the point where the lidar point to-be-colored happens to fall on the image plane.

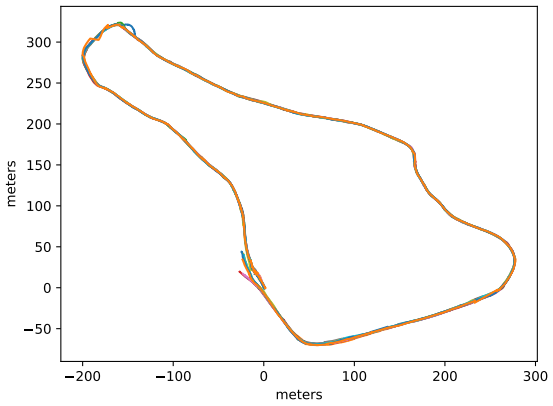
Automatic keyframe reconstruction was performed by discretizing the INS trajectory into query points. Overlapping keyframes are subsequently constructed around these points, using a fixed radius. Keyframe overlap is guaranteed by keeping trajectory discretization significantly smaller than the keyframe radius. The parameters used during keyframe reconstruction are shown in Table 2.

Parameter Name	Mapper
Voxel size	0.1 m.
Keyframe query radius	70 m.
Keyframe offset	27 m.
lidar cutoff distance	30 m.

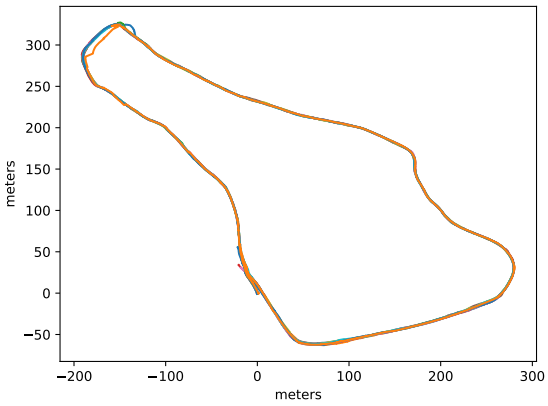
Table 2: List of Parameters used during keyframe reconstruction, minus the ICP parameters shown in Table 3.

Subsequently, a keyframe was constructed around each query point, using semantic-aware ICP, configured with the parameters and semantic weights presented in Table 3. The parameters were tuned only once, by trial and error in a challenging area where regular ICP failed to converge. The scene, located around $(lat, long) = (49.103631, 6.216095)$ is dominated by trees and includes a canopy. The same set of parameters was used for all twelve surveys. Further, semantic ICP mapping was successfully experimented in a different environment and without any change to the parameters. The environment consisted of a moving kayak inside the Orne river. The corresponding video is referenced in Section 3.6.

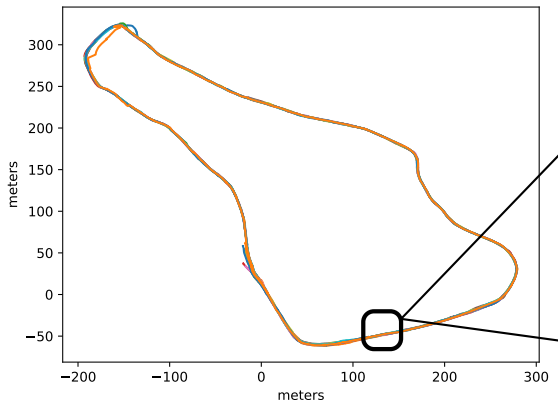
For further processing and in order to remove the overlapping parts at a later stage, keyframes and their trajectories are exported alongside the timestamps associated with every point in the cloud. We experimented with other keyframe building solutions, such as regular ICP, HDL SLAM (Koide et al., 2019) and LOAM (Zhang and Singh, 2014), with no or limited success. Although semantic-aware ICP was able to produce high quality keyframes, the choice of a keyframe builder is not critical to the proposed alignment scheme.



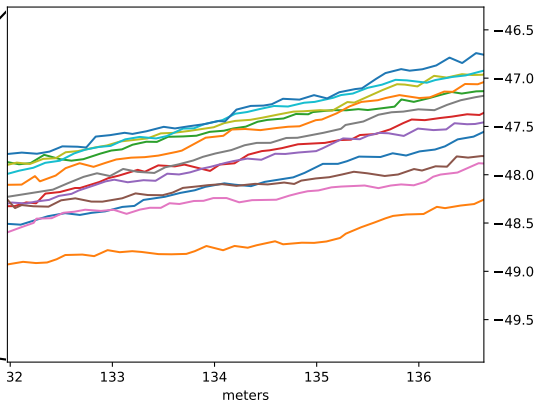
(a) Trajectory spatio-temporal alignment without loop closure and without reusing previously found transforms for overlapping keyframes



(b) Trajectory spatio-temporal alignment without loop closure, while reusing previously found transforms for overlapping keyframes (algorithm 1)



(c) Trajectory spatio-temporal alignment with loop closure, while reusing previously found transforms for overlapping keyframes.



(d) Close-up look at a portion of the trajectory. Each line represents the path taken at different surveys, around $(lat, long) = (49.1055172, 6.2165781)$

Figure 16: Qualitative assessment of trajectory alignment, a by product of keyframe alignment. Animation: here (https://youtu.be/ITfek6fPS5w).

4.3 Spatio-temporal alignment

In order to exploit the multi-dimensionality of the spatio-temporal alignment problem, temporal alignment needs to precede spatial alignment, so that to allow multiple solutions to the spatial alignment problem. This is due to the fact that spatial matches can only happen in-between consecutive, overlapping keyframes. Nevertheless, if temporal alignment precedes, it is possible to compare spatial displacement across other time periods, as detailed below.

For instance, the spatial alignment of two keyframes during t_0 can be inferred from the temporal alignment with similar keyframes taken during t_1. Further, it has been observed that keyframes are not perfectly reconstructed, even if the difference is not visible to the human eye. In a looping trajectory, this leads to error accumulation on the spatial level. Ideally, keyframes need to be wrapped both on the spatial and temporal level, in order to considerably eliminate the effects of such error, and to prevent its propagation to the temporal aspect during a global optimization effort *i.e.*, temporal links might be minimally broken to satisfy the loop closure constraint. This work however does not address this type of deformation.

The keyframes generated in Section 4.2 are then matched using ICP, using the scheme presented in algorithm 2, with a voxel size of 0.15 meters, and using the parameters presented in Table 3.

The list of transforms is then aggregated into alignment candidates according to algorithm 3. The latter is followed by the clustering of alignment candidates, as presented in Section 3.4, including loop closure.

Chosen $SE(3)$ clusters, representing chosen alignment candidates, are fed into a factor graph. The latter architecture can be represented by Figure 8, where nodes are keyframes and arrows represent the chosen kinematic links in the consensus finding step. What follows is the sparse addition of DEM priors, sparsely constrained using unary factors. Finally, point clouds are exported, meshed using LVR, and visualized using Paraview.

Parameter Name	Mapper	Matcher
Matcher	KD tree matcher	KD tree matcher
Matcher KNN size	20	2000
Max iterations	40	500
Rotation norm threshold	0.5	1
Translation norm threshold	15	45
Random input sampling probability	80%	4%
Bounding box	$-1 < x, y, z < 1$	-
Maximum input point density	10	10
Maximum ICP map point density	10	-
Semantic Paramters:		
Filtered semantic classes	Sky, person, rider, car, truck, bus, train, motorcycle, bicycle, ground	Sky, person, rider, car, truck, bus, train, motorcycle, bicycle
Semantic weights	sidewalk: 0.5, building: 1, wall: 1, fence: 0.5, pole: 1, traffic sign: 1, vegetation: 1, terrain: 1	Ground: 0.05, sidewalk: 0.5, building: 1, wall: 1, fence: 0.5, pole: 1, traffic sign: 1, vegetation: 0.75, terrain: 0.05

Table 3: List of ICP Parameters, used during keyframe generation (first column, mapping) and the spatio-temporal generation of matching candidates (second column, matching).

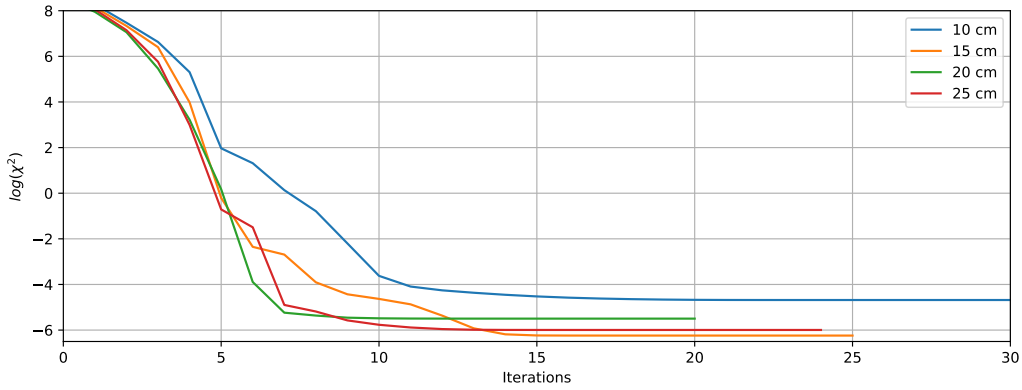


Figure 17: Evolution of the value of χ^2, a factor graph optimization residual, for voxel sizes of 10, 15, 20 and 25 cm during ICP keyframe matching. The significance of χ^2 is further discussed in Section 4.4.1.

4.4 Discussion

4.4.1 Quantitative assessment

In this section, we quantify the spatio-temporal alignment process, by analyzing the alignment quality for generated maps and their trajectories. We also discuss the chosen clustering parameters, analyse the effect of choosing a certain voxel size for matching keyframes using ICP, using optimization residuals.

Figure 11 shows a slice of 12 keyframes stacked on top of each other. The section-view shows a variation in ground level of less than 1 meter, yet not the same is happening on the tree level. This is due to the low weight given to ground points, as shown in Table 3, leading ICP to align to vegetation with varying levels.

Aligned trajectories in Figure 16d seem to fit well within the actual width of the trajectory at the query point, which has been manually measured on Google Maps around $(49.1055172, 6.2165781)$, having a value of 2.3 meters.

The effect of choosing a voxel size during keyframe matching has a visible effect on the optimization process. Although the difference is visible to the human eye, we here attempt to quantitatively compare the matching quality for different voxel sizes. To do so, a strong correlation was found between the human perception of alignment quality, voxel sizes and the value of χ^2.

The optimization residual, χ^2, describes how well a set of observations, in this paper described as constraints, fits the factor graph model, and penalizes incoherence in the implemented architecture, for example when constraints are in conflict. Therefore, in the proposed implementation, the value of χ^2 penalizes for inconsistencies originating from ICP misalignment, loop closure errors and DEM inconsistencies. By only varying voxel sizes during the matching step, the residual value of χ^2 in Figure 17 shows an improved convergence of the factor graph's underlying nonlinear least squares optimization, for values between 15 and 25 cm. The presented analysis however does not reflect on the overall alignment quality, and is only useful in comparing voxel sizes, against the overall performance of the system.

Figure 18 shows the stopping value of the clustering parameters discussed in Section 3.4. As a reminder, the stopping value is reached when at least two alignment candidates are close enough in the $SE(3)$ space to form a cluster. Still in the same figure, 91.5% of the chosen clusters had a stopping value for k_r and k_t that is smaller or equal than 0.1, whereas over 65.7% of the clusters had a stopping value, for the same parameters, that is smaller or equal to 0.05. We would like to note that values below 0.1 denote tightly knit clusters and a strong alignment consensus. Finally, 6 links were not associated with clusters, causing the associated keyframes to be discarded. The latter represents less than 1% of the 839 matches represented in

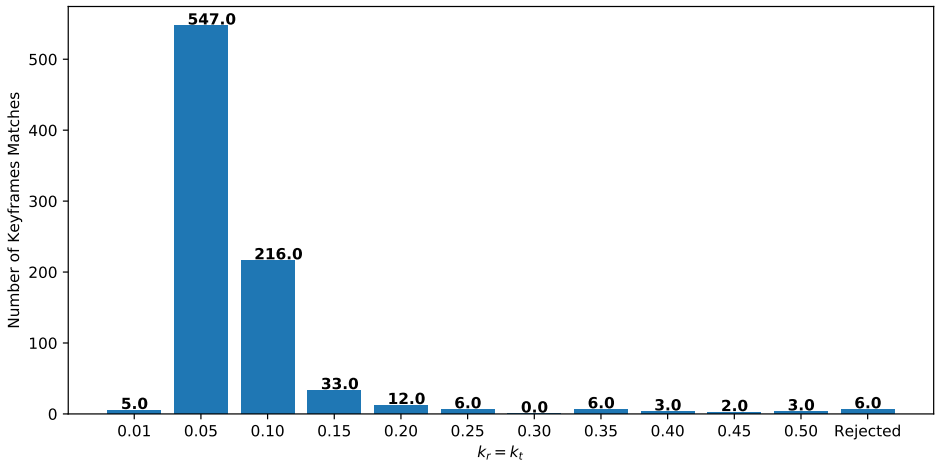


Figure 18: Distribution showing the values of clustering parameters k_r and k_t, here for chosen links only *i.e.*, after a consensus is found. The total number of matches is 839, representing 12 surveys each containing 70 keyframes, minus the loop closure link.

the same figure. Although unitless, the physical implications of k_r and k_t are described in Section 3.4.

By comparing the number of vegetation points across different months, Figure 21 shows a minimal level of vegetation cover during winter months, between December and February, with around 20% decrease by February. The trend subsequently recovers during summer months, with values pertaining to July 2021 nearly equal those captured in August 2020. The semantic class however includes all types of vegetation, including tree barks, which means that the difference between seasons could be under-estimated.

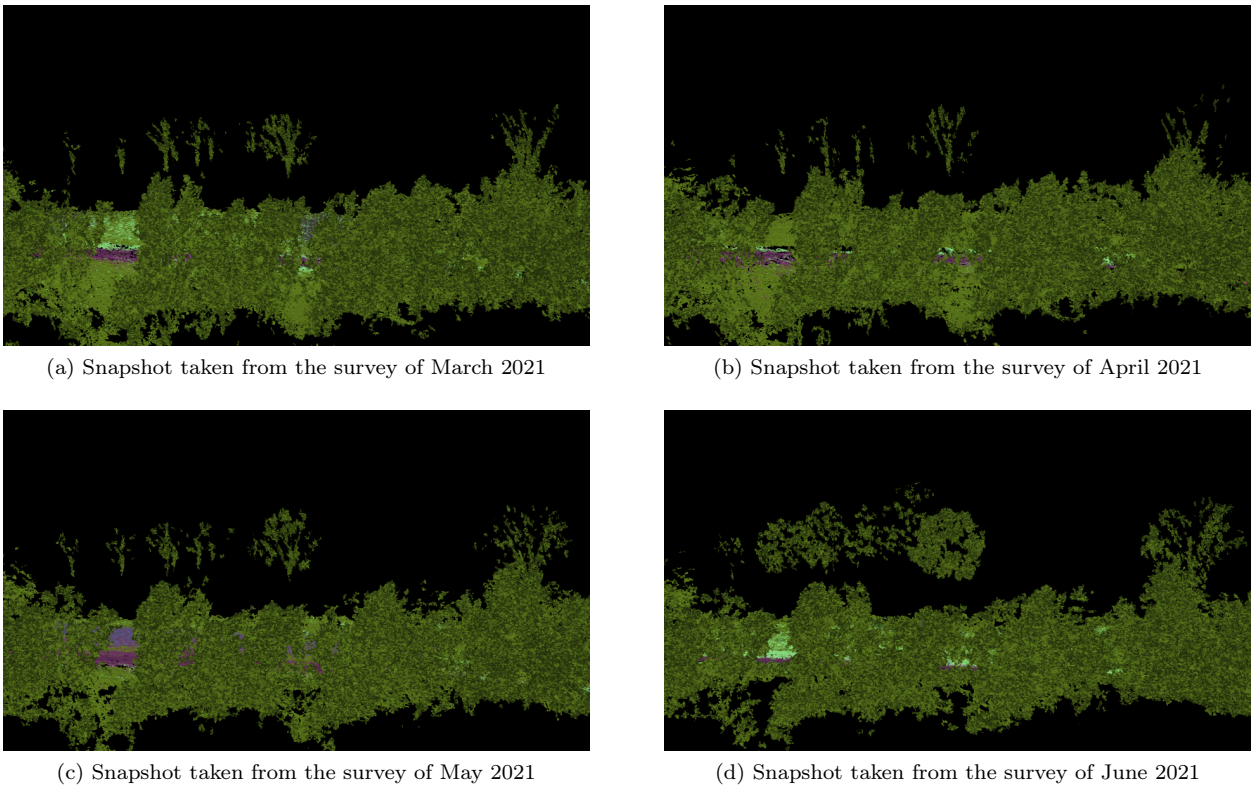
4.4.2 Qualitative assessment

Keyframe reconstruction quality was remarkably appealing to the human visual system, aided by semantic colors as shown in 9, 14 and 3. The alignment quality can be qualitatively evaluated by looking at Figure 12a. Further, a satisfactory alignment of natural environment surveys implies the possibility of perceiving slowly occurring natural changes, as shown in Figure 19.

Figure 12b shows keyframes queried at all twelve surveys, stacked on top of each other and viewed from the top. From a topological point of view, trees, the trail and the ground are easily perceivable. The accompanying video, showing the keyframes being manipulated, shows an adequate alignment from all other sides as well.

Trajectory alignment is shown in Figure 16. When compared to Figure 16a, Figure 16b shows the effect of reducing ICP runs in the overlapping parts (algorithm 1), with less drift shown in the area where the trajectory loops. Drift is further reduced after adding the loop closure constraint, as shown in Figure 16c. The difference between the latter three images is well observable in the reference video, still in the same figure.

The DEM contributed to constraining keyframe altitude. Nevertheless, since the DEM is queried using noisy GPS data, it has been observed that a sparse constraint is better suited, in order to limit drift without interfering with local estimates. This is similar to the approach used to limit drift by sparsely constraining factor graphs using GPS in (Pradalier et al., 2021) and (Chahine et al., 2021).



(a) Snapshot taken from the survey of March 2021

(b) Snapshot taken from the survey of April 2021

(c) Snapshot taken from the survey of May 2021

(d) Snapshot taken from the survey of June 2021

Figure 19: Visual perception of natural changes: time-lapse showing slowly occuring changes, a product of temporal alignment. A collection of additional time-lapses can be seen in the following video: click here (https://youtu.be/2tmxL0rMDqg).

Figure 15, and most importantly the video referenced in its caption, shows an in-depth navigation inside multiple point clouds from different surveys. The point cloud camera was constrained using a single trajectory from August 3, 2020, further adapted to constrain the both position and focal point of the virtual camera, in the other eleven point clouds.

As for semantic segmentation, It has been observed that vegetation classification is visibly robust, yet it is less reliable for ground points, probably due to the low grazing angle of the camera with respect to the ground. The proposed method however treats terrain, ground and sidewalk points as ground points. This especially true for natural environments, since ground and vegetation are of primary interest.

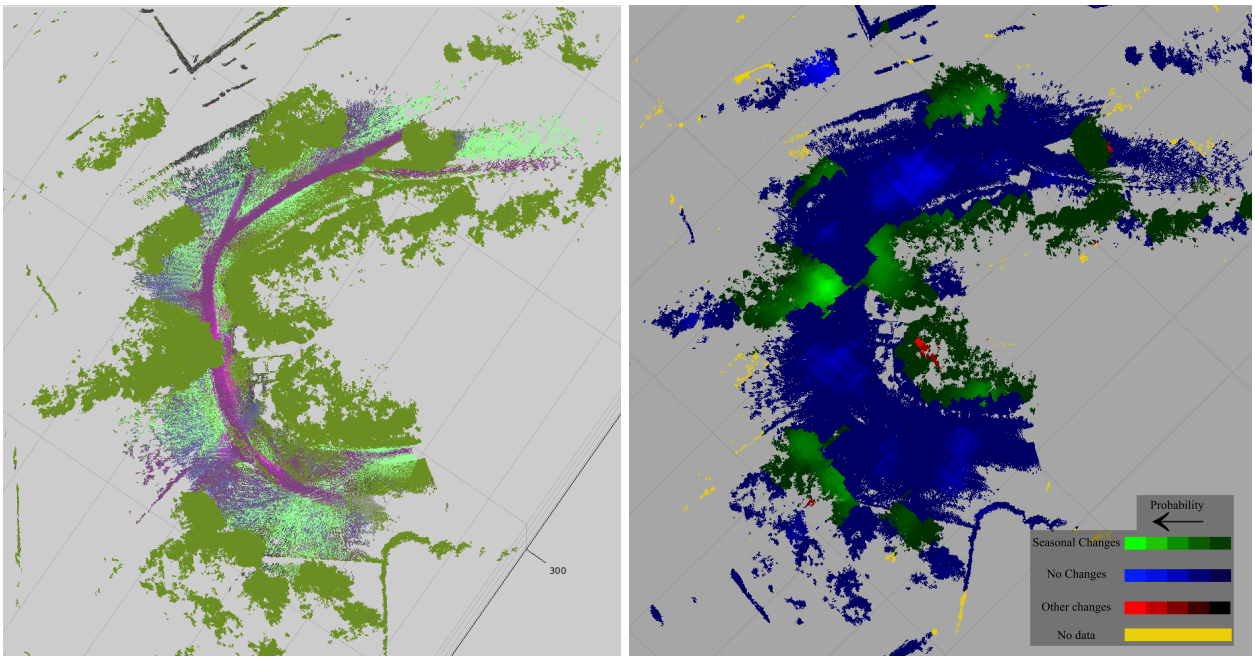
4.4.3 Ablation study

This section highlights the effects of adding semantic knowledge, ICP priors and the optimization process, by testing the proposed method with and without the latter components.

Semantic Knowledge

Without semantics, ICP keyframe building was still successful in semi-urban parts of the survey. However, ICP struggled to converge in the most natural parts of the survey, rough turns, and in places with very few features, consisting mostly of terrain and ground.

Semantic-aware ICP only failed once during all 12 surveys, on the rough turn shown on the right-most part



(a) Semantic keyframe. (b) Changes Heatmap.

Figure 20: Heatmap and semantic representations of a keyframe. Semantic knowledge was not used in the generation of the heat map, yet shown for convenience.

of Figure 22, specifically for the keyframes covering that area during the July 2021 survey. The parts that failed without semantics accounted for around 53% of the total trajectory.

INS prior for keyframe building

With semantics enabled, the effect of adding INS increments as a prior to keyframe reconstruction was found to be minimal. Without a prior, we managed to reconstruct 9 out of 10 semantic keyframes.

Without semantics, keyframe building was significantly more dependent on INS increments, failing in areas with poor GPS coverage (and therefore poor INS increments), in the most natural parts of the survey. Five out of 10 keyframes were reconstructed without semantics and without an INS prior, all of them in semi-urban areas. It should be noted that the underlying GPS-IMU Extended Kalman Filter (EKF) fusion behind the INS system does not perform as well on-foot, as it does with significant dynamics, such as when in a car. The Vectornav INS datasheet explicitly requires a speed of at least 4 m/s (15 km/h), about 3 times the average walking speed of a human, for a proper functioning of the EKF fusion.

Keyframe trajectory prior

Without an alignment prior, keyframe to keyframe matching was largely failing, even with semantics, and taking a lot of time to converge. The conclusion is consistent with the work of (Pradalier et al., 2019). A 2D trajectory prior is absolutely necessary for keyframe matching, and is more critical to the pipeline than the ICP prior for keyframe reconstruction.

The optimization effect

In this article, keyframes are manipulated as rigid bodies. As discussed in previous sections, optimization with loop closure and a DEM was found to visibly reduce overall drift. Nevertheless, a slight

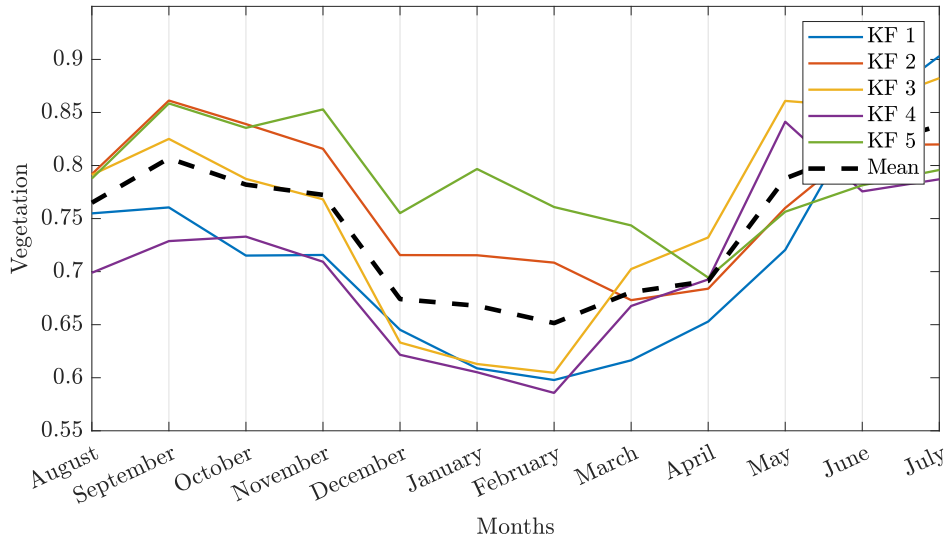


Figure 21: Percent change in the semantic label 'Vegetation' for five randomly selected keyframes, showing a minimum during winter months, between December 2020 and February 2021.

improvement in spatial keyframe stitching was found without optimization, an expected result of the rigid body assumption. In a future work, a non-$SE(3)$ optimization has the potential to address this problem.

4.5 Applications

In this section, we present sample applications that can benefit from the proposed framework. Such applications are dealing with change perception and change detection. Change perception is the ability for a human to easily perceive natural changes, by benefiting from the three dimensional alignment of surveys captured at different time periods. Change perception is however subjective in nature, and therefore falls in the qualitative category. In contrast to the latter, the quantification of natural changes aims at providing numbers that relate to slowly occurring changes in the natural environment. Finally, the purpose of detecting changes is to guide the operator to geographic points of interest, for further inspection.

4.5.1 Visual perception of seasonal changes

Changes are easily perceived in the time-lapses shown in the following video (https://youtu.be/2tmxL0rMDqg), as well as in the following video (https://youtu.be/iB0Qn1Xeuog). In the latter videos, seasonal changes are visible in the tree species that undergo major changes to foliage density. Nevertheless, an upgrade of the 16-line lidar can lead to an improved perception of small details, and would make the point cloud less susceptible to meshing artifacts. In Section 4.5.2, we present an example on how seasonal changes can be automatically detected and quantified.

4.5.2 Change quantification

In this section, we propose an example on how to detect changes in aligned keyframes, as well as areas with no or little change.

By comparing nearest neighbor points in aligned keyframes, it is expected that significant changes to tree foliage can be detected. We therefore propose to compare the z-component in-between nearest neighbors, across multiple seasons. To that end, a metric was derived: $metric = \sqrt{\sum(Z_0 - Z_j)^2}$, where Z_j is the Z component of the nearest neighbor in time period j, matched with its nearest neighbor in the corresponding

Figure 22: Red: areas where ICP keyframe building was inconsistent or failed, without semantic knowledge. Green: Areas where semantic-aware ICP was not necessary for successful keyframe reconstruction.

keyframe in time period 0 (August).

Since natural changes are local, and given the considerable size of the keyframes generated in Section 4.2, the above metric was calculated by matching nearest neighbors in 3000 5×5 m. randomly distributed patches, inside keyframes. The number of patches was chosen to be sufficiently high, as so to increase the number of points that belong to more than one patch, which allows the estimation of a classification probability, based on the number of times a point shared by multiple patches has been observed.

The first time period (August) was taken as a reference, while penalizing altitude differences with the nearest neighbor points from all the other months. Further, to eliminate noise that might affect change detection, such as occasionally accumulating snow on trees in January, the data is smoothed using a moving average, with a sliding window that spans over two months.

The formulated metric term is designed to be sensitive to seasonal changes, due to the leaf and structural growth undergone by vegetation. Additionally, seasonal changes can be modelled using sinusoidal functions (Stolwijk et al., 1999). We now propose to detect natural changes, by fitting a sinusoidal model, shown in Equation 5, to the data inferred from the changes metric.

$$model = Yscale \times sin(\frac{X - Xshift}{Xscale}) \tag{5}$$

Giving the arbitrary starting point of the dataset (August), a sinusoidal change trend that peaks somewhere around winter is expected, followed by a recovery as summer approaches. Plants also exhibit different flowering and leaf abscission periods in the spring and fall, respectively. To generalize the proposed application, we recommend a flexible sinusoidal model, that can be shifted horizontally to account for different flowering periods, and can be scaled both vertically and horizontally, to account for other discrepancies such as leaf shedding, and the period during which trees remain leafless. In Equation 5, the parameters $Xshift$, $Xscale$ and $Yscale$ are estimated using nonlinear least squares, where X is a uniform discretization ranging from one to twelve, *i.e.*, representing August to July.

An example on the quantification of seasonal changes can be seen in the sinusoidal-like trends shown in Figure 24.

(a) Snapshot from the sensor suite taken around February 2021.

(b) Snapshot from the sensor suite taken around June 2021.

Figure 23: A non-natural change to the environment. The change can be distinguished from natural changes in Figure 24.

Since natural changes follow sinusoidal trends, we expect a decent fit to the model presented in Equation 5, in the case of seasonal changes. An example of unnatural changes is shown in Figure 23, where a sports playground was erected in place of an area covered with grass. This is reflected in the value of R^2, as shown in the playground makeshift introduced after March 2020 in Figure 24a, against natural changes observed in the three other figures in Figure 24.

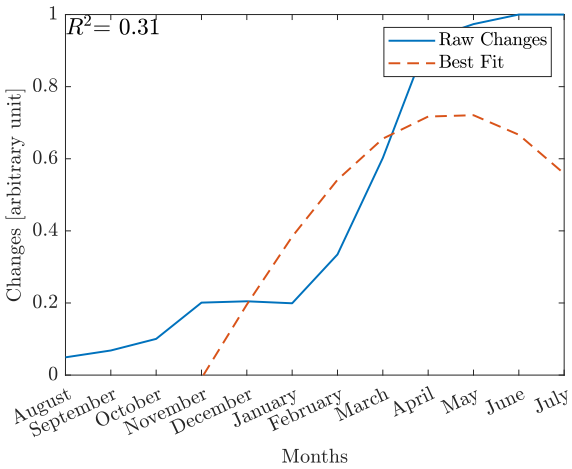
4.5.3 Heat map

In this section, we present a heat map generated from automatically detecting areas with no change, and those showing signs of natural changes. The probability field shown in Figure 20 has been inferred from recurrent observations of the same point, as discussed at the beginning of Section 4.5.

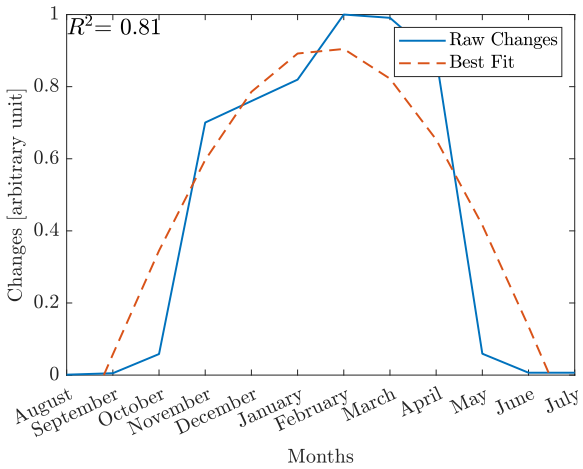
The proposed application is also capable of automatically detecting areas with no significant changes. To do so, an area is considered to undergo no changes either if it consists only of ground points, or if the same points are observed at another time period. Taking the first time period as a reference, we eliminate nearest neighbors in-between aligned keyframes. A patch is then labeled as "no change", if no or very few points remain after eliminating nearest neighbors. A second stage filter removes points that belong to a plane, estimated using RANSAC (Fischler and Bolles, 1981), to remove residual ground points, and to eliminate occasionally detected building facades. Detecting areas of no change was found to significantly reduce the chances of false positives, when it comes to detecting natural changes.

Figure 20 shows a heat map, generated using the proposed application. Still in the same figure, the vast majority of green areas, indicating seasonal changes, happen to fall on points semantically segmented as vegetation. Further, blue patches show increased confidence in areas classified as terrain, ground and sidewalk, where no changes are expected beyond grass growth.

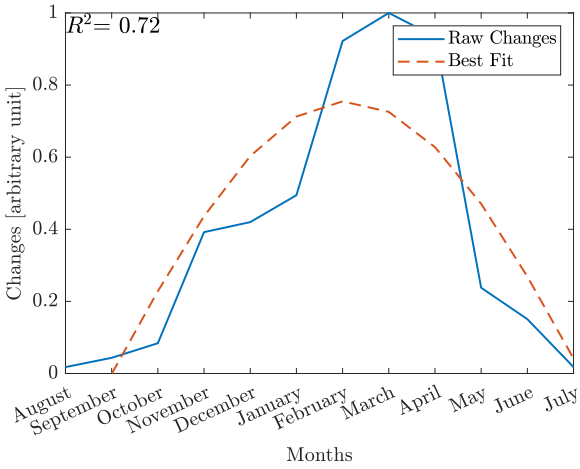
Still in Figure 20, the heat map was generated by first labelling areas with no changes, followed by the detection of sinusoidal seasonal changes. Remaining points are either not queried, shown in yellow, or shown in red to represent indeterminate changes, or false positives due to lidar coverage discrepancies in-between surveys.



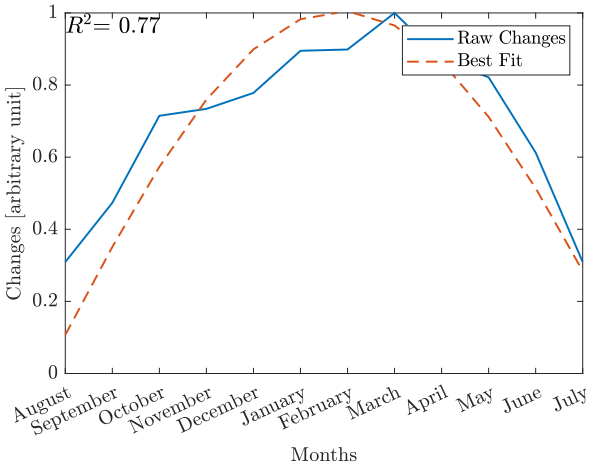
(a) Non-natural change, shown in Figure 23



(b) An example of natural changes (1/3)



(c) An example of natural changes (2/3)



(d) An example of natural changes (3/3)

Figure 24: Sinusoidal fit to the changes estimated according to the model presented in Section 4.5.

5 Conclusion

We have presented a robust, keyframe-based spatio-temporal alignment framework, for surveys captured in natural environments. It has been shown that the proposed method is capable of aligning monthly, multi-seasonal natural environment surveys taken over 12 months. Further, it has been observed that ICP can be considerably constrained using semantic priors, without the need for an exhaustive search of the parameters.

We also presented a matching scheme that organizes ICP keyframe matching. Matches were then re-arranged to form alignment candidates, with a follow-up hierarchical clustering that selects candidates with the most favorable consensus.

The proposed method is scalable in the sense that it can handle significant amounts of data, as shown in the experimental section. In addition to that, we presented two applications, that benefit from the proposed framework. The latter is however dependent on the successful fusion of data coming from multiple sources, a luxury that cannot be always afforded.

As for limitations, a very bad GPS coverage can lead to bad or no correspondence. For that to happen, a

loss of the GPS fix will have to occur exactly on a query point. A brief loss of the GPS fix outside of a query location has no to little effect on the pipeline, as it would result in the scrapping of the associated scans, that is assuming that the INS was unable to compensate for a brief loss of observations.

Future work could involve real-time change detection when mapping an already visited scene. This has the advantage of real-time change detection and the convenience of being able to inspect changes in the real world. Moreover, a deteriorated keyframe generation implies the need for additional geometric wrapping, such as in when semantics knowledge is not available. There is also a need to develop quantification tools that can further benefit from temporal alignment.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871260, as well as from the Rhine-Meuse water agency in France. We would like to thank Dr. Assia Benbihi and Laura Monnier for providing insightful feedback.

References

Ahrens, J., Geveci, B., and Law, C. (2005). Paraview: An end-user tool for large-data visualization. In *The Visualization Handbook*.

Babin, P., Dandurand, P., Kubelka, V., Giguère, P., and Pomerleau, F. (2021). Large-scale 3d mapping of subarctic forests. In Ishigami, G. and Yoshida, K., editors, *Field and Service Robotics*, pages 261–275, Singapore. Springer Singapore.

Barfoot, T. D. (2017). *State Estimation for Robotics*. Cambridge University Press, USA, 1st edition.

Besl, P. J. and Mckay, N. D. (1992). A method for registration of 3-D shapes.

Biber, P. and Duckett, T. (2005). Dynamic maps for long-term operation of mobile service robots. In Thrun, S., Sukhatme, G. S., and Schaal, S., editors, *Robotics: Science and Systems I, June 8-11, 2005, Massachusetts Institute of Technology, Cambridge, Massachusetts*, pages 17–24. The MIT Press.

Biber, P. and Strasser, W. (2003). The normal distributions transform: a new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 3, pages 2743–2748 vol.3.

Cecil C. Bridges, J. (1966). Hierarchical cluster analysis. *Psychological Reports*, 18(3):851–854.

Chahine, G. and Pradalier, C. (2018). Survey of monocular slam algorithms in natural environments. In *2018 15th Conference on Computer and Robot Vision (CRV)*, pages 345–352.

Chahine, G. and Pradalier, C. (2019). Laser-supported monocular visual tracking for natural environments. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 801–806.

Chahine, G., Vaidis, M., Pomerleau, F., and Pradalier, C. (2021). Mapping in unstructured natural environment: A sensor fusion framework for wearable sensor suites. *SN Applied Sciences*. submitted October 31.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Engel, J., Koltun, V., and Cremers, D. (2017). Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Fischler, M. and Bolles, R. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Guo, E., Fu, X., Zhu, J., Deng, M., Liu, Y., Zhu, Q., and Li, H. (2018). Learning to measure change: Fully convolutional siamese metric networks for scene change detection.

Koide, K., Miura, J., and Menegatti, E. (2019). A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement. *International Journal of Advanced Robotic Systems*, 16.

Kuemmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3607–3613, Shanghai, China.

Larsson, M., Stenborg, E., Hammarstrand, L., Pollefeys, M., Sattler, T., and Kahl, F. (2019). A cross-season correspondence dataset for robust semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Moré, J. J. (1978). The levenberg-marquardt algorithm: Implementation and theory. In Watson, G. A., editor, *Numerical Analysis*, pages 105–116, Berlin, Heidelberg. Springer Berlin Heidelberg.

Pradalier, C., Aravecchia, S., and Pomerleau, F. (2019). Multi-session lake-shore monitoring in visually challenging conditions. In *Proceedings of the Conference on Field and Service Robotics (FSR). Springer Tracts in Advanced Robotics*.

Pradalier, C., Aravecchia, S., and Pomerleau, F. (2021). Multi-session lake-shore monitoring in visually challenging conditions. In Ishigami, G. and Yoshida, K., editors, *Field and Service Robotics*, pages 1–14, Singapore. Springer Singapore.

Rosenzweig, C., Casassa, G., Karoly, D. J., Imeson, A., Liu, C., Menzel, A., Rawlins, S., Root, T. L., Seguin, B., and Tryjanowski, P. (2007). Assessment of observed changes and responses in natural and managed systems. In *Climate Change 2007: Impacts*, pages 79–131. Cambridge University Press, Cambridge, UK.

Sakurada, K., Shibuya, M., and Wang, W. (2020). Weakly supervised silhouette-based semantic scene change detection.

Schachtschneider, J., Schlichting, A., and Brenner, C. (2017). Assessing temporal behavior in lidar point clouds of urban environments. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-1/W1:543–550.

Schroeder, W., Martin, K., and Lorensen, B. (2006). *The Visualization Toolkit–An Object-Oriented Approach To 3D Graphics*. Kitware, Inc., fourth edition.

Shoemake, K. (1985). Animating rotation with quaternion curves. *SIGGRAPH Comput. Graph.*, 19(3):245–254.

Stanford Artificial Intelligence Laboratory et al. (2018). Robotic operating system.

Stent, S., Gherardi, R., Stenger, B., and Cipolla, R. (2015). Detecting change for multi-view, long-term surface inspection. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 127.1–127.12. BMVA Press.

Stolwijk, A. M., Straatman, H., and Zielhuis, G. A. (1999). Studying seasonality by using sine and cosine functions in regression analysis. *Journal of Epidemiology & Community Health*, 53(4):235–238.

Varghese, A., Gubbi, J., Ramaswamy, A., and Balamuralidhar, P. (2019). Changenet: A deep learning architecture for visual change detection. In Leal-Taixé, L. and Roth, S., editors, *Computer Vision – ECCV 2018 Workshops*, pages 129–145, Cham. Springer International Publishing.

Wiemann, T., Nuechter, A., and Hertzberg, J. (2012). A toolkit for automatic generation of polygonal maps - las vegas reconstruction. In *ROBOTIK 2012; 7th German Conference on Robotics*, pages 1–6.

Yew, Z. J. and Lee, G. H. (2021). City-scale scene change detection using point clouds.

Younes, G., Asmar, D., Shammas, E., and Zelek, J. (2017). Keyframe-based monocular slam: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88.

Zhang, J. and Singh, S. (2014). Loam : Lidar odometry and mapping in real-time. *Robotics: Science and Systems Conference (RSS)*, pages 109–111.