ISSN 1870-4069

Generation of Non-Coordinate Navigation Knowledge from a Flow of Input Views

Josué Figueroa-González, Georgii Khachaturov, Juan M. Martínez-Hernández

Metropolitan Autonomous University - Azcapotzalco, Mexico City, Mexico
{jfgo,xgeorge}@azc.uam.mx, martinez.juan.hdz@gmail.com

Abstract. An approach to automatic generation of a network of reference 'views' specifically structured for non-coordinate robot navigation is presented. The reference views are selected from the input flow while the robot moves along a random continuous trajectory in its environment. The network is organized as an atlas that represents inner model of the robot environment. It accumulates a 'rich' navigation knowledge extracted from the flow of 'poor' views. No coordinate transformation is involved, but fundamental topological concepts: the continuity of a robot trajectory and discontinuities ('heavy changes') detected in the input flow. Avoiding any explicit coordinate transformation becomes possible using a local inversion of the knowledge hidden in the relation 'robot control – change of view'. Both techniques – the inversion and detection of heavy changes – are specific for a particular robot architecture. The approach was verified on a virtual static 2D-workspace.

Keywords: cognitive robotics, learning workspace from views, non-coordinate environment modeling, visual navigation.

1 Introduction

This work is a step in a wider research about non-coordinate robot navigation. It presents a biologically inspired approach that does not use coordinates of any kind, neither for pointing a navigation goal, nor for an inner representation of navigational knowledge. 'Non-coordinate' navigation used in this work, intends to emphasize the fact that the commonly used term 'non-Cartesian' can still refer to a coordinate system. The architecture proposed here covers both, the issue of inner representation of geography, as well as learning the geography process during exploration of the environment by robot.

Fig. 1 clarifies our goal. It shows a workspace with a virtual robot. The workspace is static, bi-dimensional, and populated with obstacles. Fig. 1a shows a supervisor's view of an instance of such workspace with the robot shown as a dark triangle. If the robot had a colour camera, its current view would be as shown in Fig. 1b. But instead, robot has a sensor of 1D range images that scans all directions in a 1D field of view and measures the distance to the first obstacle along each direction, so the robot view for the same scene is shown in Fig. 1c.

Note that the distances represented by such views are not crude, but somehow transformed due to technological reasons related to OpenGL.

The single view in Fig. 1a provides a human with full information about geography of the environment, but the robot, at any time, has a much poorer input: the one similar to the shown in Fig. 1c. By processing a flow of such 'poor' views, is constructed a data structure that contains a complete information about the geography, equivalent to the one in Fig. 1a.

The inner workspace model of the robot is constructed as a set of reference views from the input flow and is organized as an atlas of 'neighbourhoods'.

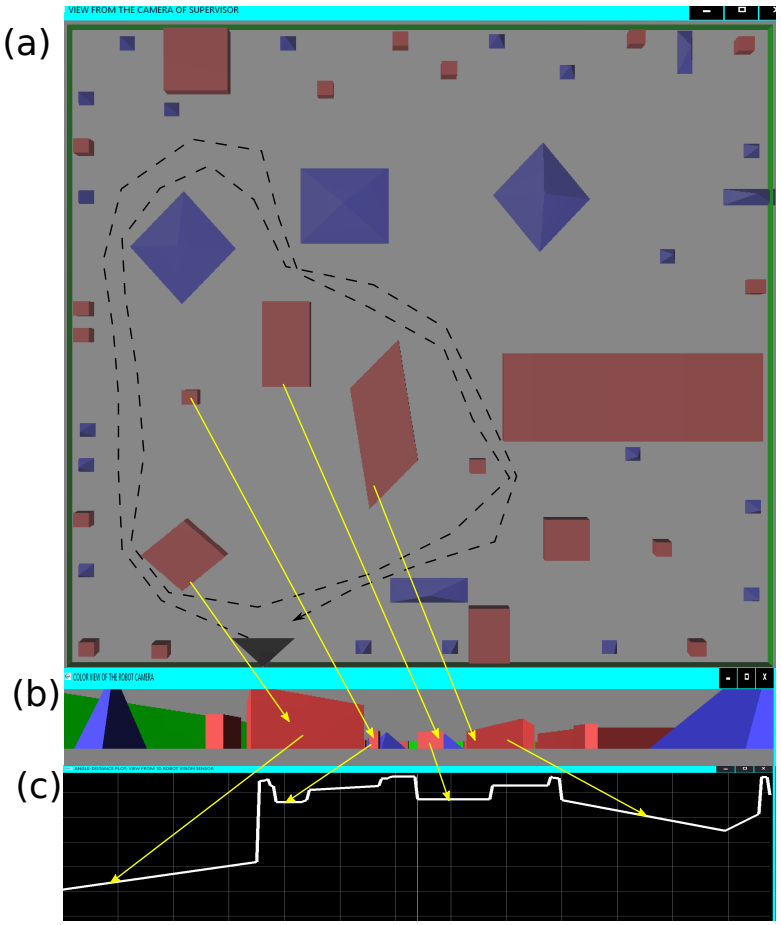

Fig. 1. Three views for the same "robot-in-environment" scene: (a) A supervisor's view; the robot position and orientation is shown by a dark triangle at the bottom; the dashed line is commented in Section 4. (b) A robocentric view by a colour camera. (c) A robocentric view from a sensor of 1D range image. The lines between (a) and (b), and between (b) and (c) connect the same objects on corresponding views.

The process of learning the geography is based on driving the robot along a continuous random trajectory and an analysis of the input flow of such views; at the beginning, the inner model of workspace is initialized to a single reference view that corresponds to an initial robot position on the scene. A test over any pair of views, being applied to the pairs of adjacent views of the input flow, is aimed to show semantic discontinuities in the flow. The positive responses of the test are called 'heavy changes'. Each neighbourhood is formed by a set of views with no heavy change between them; the data structure that represents a neighbourhood contains also information about entrances from and exits to other neighbourhoods. This network is a dynamic structure: it is subjected to the inclusion of new reference views, linking them to neighbourhoods, adding new neighbourhoods into atlas, adding links (entrances or exits) between adjacent neighbourhoods and merging neighbourhoods after detecting a loop closure of the robot trajectory in its workspace.

The network represents a graph-based component of the robot navigation knowledge. It can be used for solving non-local navigation tasks, meanwhile for the local navigation a vision-guided control is assumed. Retrieving a command for a desirable change of views is performed by the Kohonen associative memory, [15]. Training the memory is accomplished by a set of 'cause–effect' pairs {robot command; change of views}.

Navigation assumes self-localization, in contrast to SLAM, localization in the scope of presented approach is non-coordinate; it is performed as a permanent task that at any moment generates a pair {Current Neighbourhood (CN); Current Reference View (CRV)}, where the components, respectively, stand for IDs of the neighborhood which the current robot view belongs, and a reference view in CN from which the current view can be reached by a robot control command. An estimate of the command is constructed by associative memory.

For the sensor that produces images like the ones in Fig. 1c, estimation of the robot control command for a desirable change of views is as follows: The associative memory is represented as a vector with components indexed by possible distances to objects in a view. The vector representation is required because each object in the field of view responds non-linearly to a robot control according to the distance to this object. For two views that determine a required change of robot state, any pair of matching objects in them contributes with its own impact into the estimate of the command. This impact is computed by a component of the vector of associative memory that corresponds to a distance to this particular object. Amount of all impacts gives the required estimate.

Heavy changes are the milestones for the non-coordinate navigation. Detection of heavy changes is sensor-dependent. A corresponding technique for the views like in Fig. 1c was presented in [14]. The relation of heavy change helps to associate the robot views with some neighborhoods of the atlas: two close views generated on a continuous robot trajectory belong to the same neighborhood if there is no heavy change between them, and vice versa.

This work focuses on the logic of creating the network of reference views, this approach was tested using a new version of the software presented in [14].

The rest of the paper is organized as follows: Section 2 offers a review of previous works. Section 3 describes a logic for learning the inner model of workspace. Section 4 describes the structure of software and experiments on it. Section 5 contains conclusions.

2 Related Works

2.1 Non-Coordinate Navigation

The attempts to understand how the human brain tackles the navigation tasks have been undertaken in numerous works during several decades, mainly in psychology. In robotics, it is an active theme for more than two decades.

An approach to navigation of autonomous systems in semi-structured environments using natural language is based on a model called Generalized Grounding Graphs (G3) [19]. The robot must be trained beforehand for recognizing basic elements like: events, objects, places and paths. Then they can be used to express a navigational goal in natural language as "go to next corner of the room". Closer ideas can be found in [2], where the robot must complete a task which involves identifying an object and a position in an unknown environment. Once the task and object are identified, the robot must go to a specific location. In this work the robot learns using imitation learning. In both works, navigation does not deal with a coordinate space.

A graph based model for a non-coordinate vision-guided navigation was proposed in [13]. Other works on this topic are [8,9,18]. The model presented in [13] (so called GT-model) is formed by two graphs: G, which stands for the robot vision-to-action knowledge, and T that represents a geography or administrative system imposed on G. G alone is quite similar to the view graph of [8,18], but being enriched by geography graph T it obtains some new properties.

In formal terms, a GT-model is defined as pair {G, T}, where the components satisfy the following properties:

- G = (V_G, E_G) is a directed graph with non-negative weights assigned to its edges; intuitively, any node of V_G stands for a set of close robot views and implicitly it represents a subset in a space of all robot states; the edges of E_G stand for some control rules that drive robot from one set of close views to other.
- T = (V_T, E_T) is a directed graph with an unique source-node whereas the set of its terminal nodes coincides with V_G; the nodes of V_T stand for geographic entities of the robot space, whereas the edges of E_T are just inclusions indicating that one entity belongs to some other.

In these terms, for example, "Move robot from the Mexico City downtown to the Computer Research Center", means to drive robot from one set of views (downtown) to another (CRC) by means of the rules from E_G. Then, a formalization of a non-coordinate navigation problem is as follows. Let α, $\beta \in V_T$ be geographic entities and V_α, $V_\beta \subset V_G$ be their respective domains. The problem

is: Find a path in G with minimal summary weight that begins inside V_α and ends inside V_β. This problem is called the extended shortest path problem (*espp*). So an *espp* means search of a best route that connects two sets from V_G, however not arbitrary ones but only those represented by some nodes of the geographic graph T.

It turns out, [13], that a navigational problem in the form of *espp* under some natural assumptions can be solved by a dynamic programming algorithm of lower computational complexity than for a classic shortest path problem for two nodes of graph G.

In spite of some advantages of the approach based on GT-model, its application to robotics for the years after publication of [13] remained as an abstract theory than a practical instrument due to the difficulties related to learning the lowest, non-verbalizable level of geography.

Indeed, consider a geographic item that has an explicit name like North America, Mexico City, The National Polytechnic Institute of Mexico, etc. It is technically clear how to introduce such an item into a GT-model. But for the lowest geographic level, each item should be represented by a data structure that does not have neither a distinctive shape, nor a name. In this respect, this work together with [14] represents a novel step to overcome obstacles for future practical implementation of the non-coordinate approach to navigation of [13].

Note that in contrast to the verbalizable levels, the lowest-level geography model is sensor-dependent: the higher levels can be the same, say, for a robot, a blind person, as well as for a human who does not suffer any illness of sight; the lowest level geography depends: for a blind person – on the stick which he/she uses for exploration of the environment, for a normal person – on his/her vision. This is why learning geography of the lowest level cannot be done in the same way for all kinds of robots. But some common principles can be found.

2.2 SLAM and the Navigation Based on a Sketch of Path

The research line called Simultaneous Localization and Map Building (SLAM) has a story of more than a three decades [4–6, 11]. The aim of this approach is to convert a set of robocentric views into a single map. This 'map' is understood in a traditional, Cartesian meaning so that a human could read and use it. The SLAM techniques involve statistical methods including extended Kalman filter [3] and Rao-Blackwellized particle filters [10]. They allow feeding the map creation while the robot moves smoothly.

An approach to robot guidance in an indoor environment is presented in [1]. As a human-robot interface, it makes use of a sketch of a required robot path; the robot task in a sense is an inversion of SLAM: starting from a poor quality map (sketch), identify its landmarks in the robocentric views, and use them for guidance. The sketch can be regarded as a single-task-oriented, "light" navigational model of environment. The recognition is simplified due to the indoor specific.

We adopt here the, employed in many SLAM works, idea about treating loops, [7, 12, 17]. Detection of a loop-closure in a robot trajectory increases

drastically the precision of the map under construction. In the scope of our approach, detected loops allow to improve adequateness of an inner model of the workspace.

Note that a single 2D view in Fig. 1a is equivalent to a 2D map, however this way of presentation of the workspace is chosen here not because of some limitations of our approach but just for illustrative and testing purposes. A map like in SLAM not always allows an adequate representation of geography of a robot environment. For example, it is inadequate for an environment similar to a 3D-maze, or for a mountain cave of a complex structure. On the opposite, the approach presented in this work is not limited to a 2D-environment.

Nevertheless, we believe that ideas of SLAM are very fruitful to be combined in future with our approach especially for facing the robot dynamics that are disregarded yet in our experiments.

2.3 Detection of Heavy Changes

Robot state for the workspace of Fig. 1 can be regarded as the pair {robot position; robot orientation}. From a topological point of view, while robot moves smoothly in workspace, a path in the space of robot states is mapped continuously into a space of robot views. Some qualitative changes in the flow of views are declared as Heavy Changes (HCs). Detection of HCs helps to organize the inner model of workspace. As it was mentioned in section 1, a particular technique for detection of HCs depends on the robot architecture. What a HC means depends also on a semantic interpretation of objects in a current view: for instance, in the flow of range images like Fig. 1c, the dynamics of distant objects is not as relevant for HC as that of close objects. Then, detection of HCs in a border zone between different neighbourhoods provokes a natural instability of outcome. In these circumstances, providing the robustness of the atlas of neighbourhoods formed by reference views is not an elementary issue.

A method for tackling this issue was presented in [14] for the virtual world of Fig. 1. There the flow of 1D range images is subjected to the processing that combines a transformation of each image into a string of symbols of an alphabet with comparison of such strings by means of a technique based on the Levenshtein distance [16]. Note that the Levenshtein distance is known nowadays mainly for its role in detection of plagiarism.

Any symbol of the string is constructed as a singular point, that is as a discontinuity of the first or second derivative of the range image; the alphabet of singular points to code a range image includes {STEP UP, STEP DOWN, ANGLE TOWARD ROBOT, ANGLE OUTWARD ROBOT} with obvious intuitive meaning for each option. Since this alphabet is rather short for reliable comparison of the strings, each symbol is supplied with an extended descriptor to reduce the probability of errors while taking a decision on occurrence of a HC.

Computation of the Levenshtein distance for two strings leads to so called Editorial Prescription (EP) that describes a simplest transformation of one string into another by means of some elementary operations over strings. If, for example, EP is empty then the strings are equivalent and there is no HC

between corresponding views; otherwise a decision on occurrence of HC is taken by a deeper analysis of EP. We refer the readers to [14] for further details.

Note that this approach cannot be applied directly to 2D images because the technique of Levenshtein is oriented exclusively toward processing the strings, so in general the detection of HCs is an open field for future research.

3 Inner Model of Environment

We start with some preliminary comments on the kind of robot sensor.

In experiments we deal with 1D range images, however the term 'views' does not refer to this particular kind of sensor. Note that, although different kinds of sensors generate views that seem qualitatively different, sometimes their information impact is almost equivalent. For instance, conventional video snapshots like in Fig. 1b can be chosen as an input flow and although such input seems rather different from the range images, the extraction of the range data virtually equivalent to that represented in Fig. 1c can be easily performed by processing the colour views as the robot moves in its workspace.

It is quite reasonable not using crude but somehow post-processed views to make them more robust about noise, illumination, etc., as well as more compact for storing. For example, instead of the crude range images similar to Fig. 1c, the strings described in section 2.3 are stored in our experiments.

The network of reference views mentioned in section 1 represents an inner robot model of its environment. It is constructed as a set of robocentric views specifically organized to meet navigation needs. The inner model comprises two databases: VIEWS that stores the set of reference views and ATLAS that imposes a kind of topology on VIEWS and represents them in the form of a set of neighbourhoods.

Each neighbourhood comprises three sets: the set $\{b_1,b_2,...,b_n\}$, where each b_i can be regarded as ID of a Reference View (RV) in VIEWS, the set of entrances, and the set of exits.

The entrances and exits have the same format of pair {neighbourhood_ID, RV_ID}, where the IDs refer to respective objects in ATLAS and VIEWS. For an entrance, values of the components specify such objects *from which* the robot can be moved to the Current View (CV) by a command, whereas for an exit they stand for the objects *to which* the robot can be moved starting from a view b_i that belongs to the neighbourhood that owns the exit.

In its turn, the correspondence of a robot position to a neighbourhood of ATLAS is the matter of the self-localization process mentioned in Section 1. Mathematically, self-localization for non-coordinate navigation can be regarded as correspondence: CV $\longrightarrow$ {CN,CRV}.

All transformations of the inner model are carried out by a logic triggered just at the moments when this correspondence fails to be established. The remainder of this section presents this logic.

In a general case, CV does not belong to VIEWS, and hence it may not belong to the set $\{b_1,b_2,...,b_n\}$ of CN. However for a correct self-localization in

a workspace, the reachability of CV from a reference view b_i is required. Under the assumption that the robot state in the CN yields view b_i the reachability means that, a single robot control command exists that moves robot from b_i to the state at which the CV has been produced, in other words, to the current state.

It is the task of the associative memory mentioned in Section 1 to construct an estimate of such command starting from a view b_i and CV. If the norm of the estimate is greater than a threshold, such estimate is rejected. If the same is true for all $b_i \in \{b_1, b_2, ..., b_n\}$, the hypothesis of reachability of CV from CN is rejected as well.

In the process of learning the inner robot model, supervisor randomly generates a succession of robot control commands. Robot executes each command and gets a new CV; the new CV is subjected to analysis that leads to actualization of self-localization pair {CN; CRV}, as well as to possible modifications of both databases, ATLAS and VIEWS.

In this process, outcome of the following two tests completely determines possible modifications of the model: the Test on Reachability (TR) of CV and the Test on a Heavy Change (THC) between the previous view and CV.

A negative response of THC (no heavy change) means that CV corresponds to the same CN as the previous one, and the set of neighbourhoods of ATLAS remains the same. However, the result of TR is still important: VIEWS stays unchanged for a positive response, but otherwise the previous view of the input flow is added to VIEWS and included as a new element into set $\{b_1, b_2, ..., b_n\}$ of CN. Since CV has been just reached from the previous view by the current robot command, this inclusion automatically provides the required reachability, and the just added view becomes a new CRV.

For a positive response of THC (heavy change occurred) further logic follows the block-diagram of Fig. 2: it starts from a straightforward search of an exit among those already registered in CN. If an appropriate exit is found, then CN must be substituted with the neighbourhood that the exit leads to. Just as in the previous paragraph, actualization of CRV depends on the response of TR: a positive response of TR is automatically accompanied with new CRV, whereas for a negative response, CV is included into VIEWS and added to the set of views of the actualized CN and CV becomes a new CRV since a view is reachable from itself.

If the search of exits fails, still a chance remains that robot has completed a long loop and returned back to an old neighbourhood. So search of loop closure is executed: CV is subjected to THC and TR with respect to the views of old neighbourhoods. For closure, it is necessary to find an old view with a negative response from THC and a positive from TR. Unfortunately, this condition is not enough to avoid errors completely: similar views can be perceived as distant states of robot. To reduce the probability of these errors, too uniform views can be filtered out from the search as unreliable; next, the sensor system of the robot can be enriched by a virtual compass (a sensor of orientation), which allows filtering out the candidates with a strongly different orientation than a

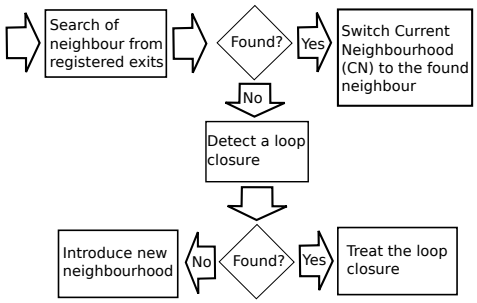

Fig. 2. Block-diagram of transformation of the model after a heavy change.

current robot orientation. As an ultimate resource, robot can call supervisor for help in doubtful cases.

If the search for loop closure fails, a new neighbourhood is introduced into ATLAS and the CV becomes its unique view. A link is established between the new neighbourhood and CN that is an entrance and exit are introduced into respective neighbourhoods; and then CN is actualized to be the new neighbourhood.

Otherwise, if a loop is detected it is treated as follows. First, it is clear that the CN turns out to be a new entrance to an Old Neighbourhood (ON), on which the loop closure has just been detected. Hence a corresponding link – new entrance to ON, new exit to CN – is introduced, but it is possible that this entrance is not brand new because it had been introduced earlier to ON under a different ID. To study this possibility, a view b_i from CN is subjected successively to THC with respect to the views of all entrances to ON. If an entrance E is found, whose view component yields a negative response of THC, the hypothesis is accepted that the neighbourhood component of E is identical to CN is accepted. Since the views under consideration are topologically close each other, although the errors of identification of views are still possible, their probability is much less than for loop closure. As a consequence, CN and the neighbourhood indicated in E can be merged into a single neighbourhood; depending on which neighbourhood is chosen to absorb the other, one of the two neighbourhoods can be deleted from ATLAS after merging. It also implies substitution of all references in ATLAS to the deleted neighbourhood by the ones to the remaining neighbourhood.

The previous paragraph, in a simplified form, presents the idea of a surgery of the inner model after a loop detection. In fact, a detected loop allows a deeper surgery to be applied. Indeed, THC can be applied, for instance, to the entrances of CN versus the entrances of entrances of E. Furthermore, the same idea can be extended to the exits. It would help to establish more equivalent neighbourhoods additionally to those of the previous paragraph.

Applying a deeper surgery makes the inner model to converge faster, but even the simple surgery leads to the convergence of ATLAS: the model tends to stabilize itself as the robot route becomes sufficiently dense in the workspace.

4 Experiments

A newer version of the pilot software described in [14] was used for the experiments. The functions implemented in the package include:

- Interactive creation of obstacles of the robot environment.
- Visualization of the environment and different kinds of robot views.
- Intuitive supervisor graphic interface for controlling robot.
- Training the Kohonen associative memory.
- Transformation of angle-distance plot to a string of singular points.
- Application of a technique based on the Levenshtein distance to the strings of singular points for detection of a heavy change.
- Auto-localization process.
- Manipulations with the inner model of the environment.
- Serialization/Deserialization.

Using software comprises three stages: generation of obstacles inside the workspace; training the associative memory and learning geography of the workspace.

For a generated workspace, just after the Kohonen associative memory has been trained, the program automatically initializes: the robot position shown by the triangle in Fig. 1a; the content of ATLAS of a unique neighbourhood and the content of VIEWS to a unique view like the one in Fig. 1c.

Supervisor drives robot so that each mouse click is interpreted as a leading point to be translated in a combination of rotation $< 4°$ and displacement ahead $< 3\%$ of the workspace size. About 80 commands are required to close a loop shown by dashed line in Fig. 1a. The processing of each command of the sequence follows the logic presented in Section 3.

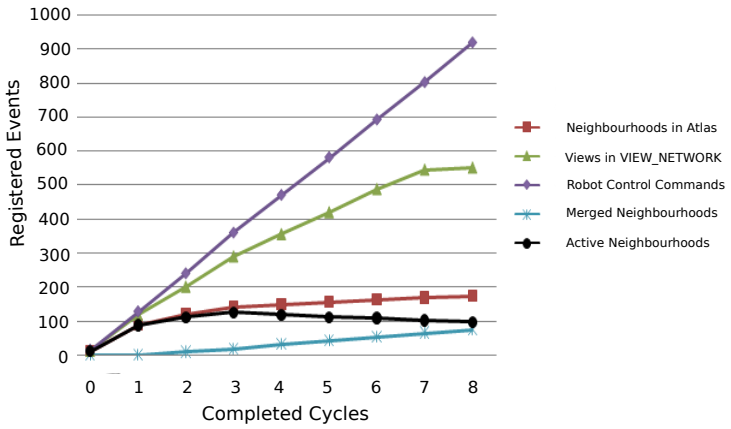

Fig. 3. Plot of statistics obtained during experiments.

A typical experimental result of learning the workspace is illustrated by Fig. 3. It presents several statistics that show the dynamics of generation of the inner model. The statistics are registered for a looped route that approximately follows the dashed line of Fig. 1a. Supervisor leads the robot several times along the loop, and the statistics are monitored as functions of the number of full revolutions.

The plotted statistics include the number of: (i) commands; (ii) neighbourhoods introduced into ATLAS; (iii) merged neighbourhoods; (iv) views introduced into VIEWS. One additional plot shows the number of active neighbourhoods in ATLAS defined as 'introduced neighbourhoods minus merged neighbourhoods' that is the difference (ii)–(iii). This plot shows an obvious tendency for stabilization of the fragment responsible for the loop in question in the model under construction.

5 Conclusions

The presented approach for a robot learning from its workspace employs a graph-based model ideologically similar to the view graph, [8, 18], or the graph G of the GT-model, [13]. The inner model of workspace is constructed as a set of reference views from the input flow, organized in an atlas of neighbourhoods. Our approach is parallel to SLAM, but no coordinate transformation is involved and no map of the workspace is build: the intention to avoid any coordinate representation is the main bio-inspired motivation for this work.

Instead of a explicit use of coordinate transformation we deal with its implicit substitute based on a local inversion of the cause-effect correspondence between a robot control and a sensor response; to provide this inversion, a Kohonen associative memory is used.

This work focuses on the logic of transformation of the inner model: adding new reference views; agglomeration them as neighbourhoods; switching a current neighbourhood; adding a new neighbourhood into the atlas and merging some neighbourhoods detecting a loop closure on a robot trajectory. Two tests govern learning the inner model: test on reachability of a current view from a reference view, and test on a semantic similarity of two views. The former decides whether a control exists that would provide a needed change of view; the latter determines if robot still belongs to a current neighbourhood. Only similar reference views (no 'heavy change' between them) constitute a neighbourhood.

The logic of creation of the inner model was tested on a virtual robot in a 2D workspace. The flow of input views in this case is represented by 1D-range images. For this kind of virtual sensor the detection of heavy changes can be effectively done, [14], by transforming each view to a string of singular points and a subsequent comparison of such strings by a technique based on application of the Levenshtein distance. The experiments carried out for this work give a statistical confirmation of stabilization of the inner model of the workspace when supervisor leads the robot many times along a cyclic trajectory.

This work proposes a novel method for automatic construction of a graph-based model of workspace, formed by reference views. Using such models opens

the way to solve navigation problems in natural terms. It can be done, [13], by introducing an additional structure to represent the ordinary geographic concepts. This issue presents a future research lines. Other lines for future works are: replacement of the 1D robot sensor with other kinds of sensors; extension of the technique from the 1D to 2D views, as well from a static to non-static workspace, and more. As a challenging goal oriented to practice, we can mention the development of a drone able to navigate automatically in a region after learning it just by flying several times over it.

References

1. Boniardi, F., Valada, A., Burgard, W., Tipaldi, G.D.: Autonomous indoor robot navigation using a sketch interface for drawing maps and routes. In: Robotics and Automation (ICRA), 2016 IEEE International Conference on. pp. 2896–2901. IEEE (2016)
2. Boularias, A., Duvallet, F., Oh, J., Stentz, A.: Grounding spatial relations for outdoor robot navigation. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on. pp. 1976–1982. IEEE (2015)
3. Castellanos, J., Martinez-Cantin, R., Tardós, J.D., Neira, J.: Robocentric map joining: Improving the consistency of EKF-SLAM. Robotics and autonomous systems 55(1), 21–29 (2007)
4. Cheeseman, P., Smith, R., Self, M.: A stochastic map for uncertain spatial relationships. In: 4th International Symposium on Robotic Research. pp. 467–474 (1987)
5. Davison, A.J., Murray, D.W.: Simultaneous localization and map-building using active vision. IEEE transactions on pattern analysis and machine intelligence 24(7), 865–880 (2002)
6. Dissanayake, M.G., Newman, P., Clark, S., Durrant-Whyte, H.F., Csorba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on robotics and automation 17(3), 229–241 (2001)
7. Eade, E., Drummond, T.: Unified loop closing and recovery for real time monocular slam. In: BMVC. vol. 13, p. 136. Citeseer (2008)
8. Franz, M.O., Schölkopf, B., Mallot, H.A., Bülthoff, H.H.: Learning view graphs for robot navigation. Autonomous robots 5(1), 111–125 (1998)
9. Gomi, T.: Non-cartesian robotics. Robotics and Autonomous Systems 18(1-2), 169–184 (1996)
10. Grisetti, G., Tipaldi, G.D., Stachniss, C., Burgard, W., Nardi, D.: Fast and accurate SLAM with Rao–Blackwellized particle filters. Robotics and Autonomous Systems 55(1), 30–38 (2007)
11. Guivant, J., Nebot, E., Baiker, S.: Autonomous navigation and map building using laser range sensors in outdoor applications. Journal of robotic systems 17(10), 565–583 (2000)
12. Kähler, O., Prisacariu, V.A., Murray, D.W.: Real-time large-scale dense 3d reconstruction with loop closure. In: European Conference on Computer Vision. pp. 500–516. Springer (2016)
13. Khachaturov, G.: An approach to trip-and route-planning problems. Cybernetics and Systems 33(1), 43–67 (2002)

14. Khachaturov, G., Figueroa, J., González, S., Martínez, J.M.: Heavy changes in the input flow for learning geography of a robot environment. In: Lecture Notes in Artificial Intelligence. pp. 518–529. Springer (2017)
15. Kohonen, T., Lehtiö, P., Rovamo, J., Hyvärinen, J., Bry, K., Vainio, L.: A principle of neural associative memory. Neuroscience 2(6), 1065–1076 (1977)
16. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet physics doklady. pp. 707–710 (1966)
17. Mei, C., Sibley, G., Cummins, M., Newman, P., Reid, I.: Rslam: A system for large-scale mapping in constant-time using stereo. International journal of computer vision 94(2), 198–214 (2011)
18. Schölkopf, B., Mallot, H.A.: View-based cognitive mapping and path planning. Adaptive Behavior 3(3), 311–348 (1995)
19. Tellex, S., Kollar, T., Dickerson, S., Walter, M.R., Banerjee, A.G., Teller, S.J., Roy, N.: Understanding natural language commands for robotic navigation and mobile manipulation. In: AAAI. vol. 1, p. 2 (2011)