Autonomous Indoor Robot Navigation Using Sketched Maps and Routes

Federico Boniardi Abhinav Valada Wolfram Burgard Gian Diego Tipaldi
Autonomous Intelligent Systems Group, University of Freiburg, Germany
{boniardi, valada, burgard, tipaldi}@informatik.uni-freiburg.de

***Abstract*—Hand drawn sketches are natural means by which a high level description of an environment can be provided. They can be exploited to impart coarse prior information about the scene to a robot, thereby enabling it to perform autonomous navigation and exploration when a full metrical description of the scene is not available beforehand. In this paper, we present a navigation system supplemented by a tablet interface that allows a user to sketch a rough map of an indoor environment and a desired trajectory for the robot to follow. We propose a novel theoretical framework for sketch interpretation based upon the manifold formalism in which associations between the sketch and the real world are modeled as local deformation of a suitable metric manifold. We also present empirical results from experimental evaluations of our approach in real world scenarios both from the perspective of the navigation capability and the usability of the interface.**

I. Introduction

The design and implementation of intuitive methods of communication between robots and humans has attracted considerable amount of attention from both the robotics and AI communities thus far [19]. In the context of robot navigation, significant amount of research has been devoted to develop natural and human-friendly means for transferring spatial information from users to robots as well as to enhance the robot's cognition about the surrounding environment [24, 23, 6, 12]. The ability to navigate in a known environment is a key requirement for robots to be fully autonomous. Such a capability is usually achieved employing metrically consistent maps, which are retrieved up front typically by means of human teleoperation or autonomous exploration. Although many popular methods for simultaneous localization and mapping (SLAM) have proven to be extremely efficient as well as accurate [5, 4], they all require preliminary operations that could be tediously time-consuming or sometimes even unfeasible. Rescue scenarios, for instance, are common examples where remotely controlling a robot could be impossible for an external operator. Furthermore, new service applications require robots to be employed even by naïve users, such as older people or children, which would be overburdened by tedious or excessively complex operations.

To overcome these difficulties, researchers have investigated the use of hand-drawn maps and sketches to provide a rough descriptions of the environment. An early attempt to perform simple navigation tasks only relying upon sketched maps was suggested in [8]. In this research, the authors proposed a POMDP based approach to learn a metrical conversion

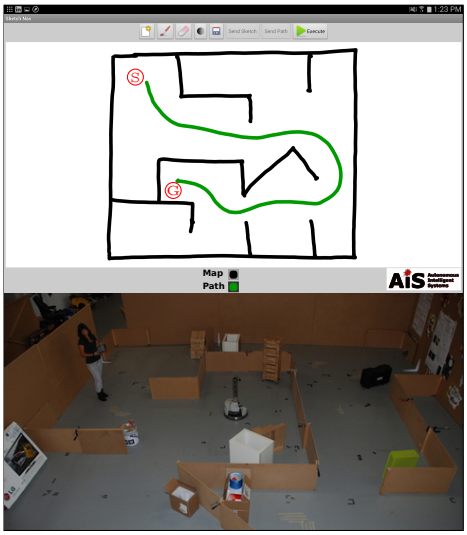

Fig. 1. Top: Snapshot of the tablet interface. Figure shows the drawn map and the path (green). The starting (S) and goal (G) positions are annotated. Bottom: The environment used for the experiments.

between a sketch, encoded as a topological map, and the real world. More recent approaches have tackled the problem of providing a quantitative interpretation of a hand-drawn sketch via landmarks' matching, mimicking human-like navigation. Kawamura *et al.* [7] developed a full navigational system in which a robot is instructed to track a trajectory in a sketch. The robot navigates heading towards the waypoints that best match the predicted scenario perceived by the robot's sensors and the landscape observable by the waypoints. The current robot pose is meanwhile tracked by triangulating the relative positions of the predicted landmarks.

A wide and deep investigation into sketch-based navigation has been proposed by Skubic *et al.* [19, 15, 16, 17, 18, 3]. In their works, the authors focused on designing and testing sketch interfaces with the aim of instructing a robot, or even a team of robots, to perform simple navigation tasks. The users were required to sketch a map of the scene and a feasible path. A fuzzy state controller is then responsible for outputting suitable motion commands based on the qualitative state of the robot inferred from local sensor readings. The state is retrieved from the spatial relations between landmarks, modeled using histogram of forces, and later converted in a linguistic description by means of fuzzy rules. Shah and Campbell [14] have proposed an extension to this approach. The authors used techniques inspired from landmark-based SLAM to track

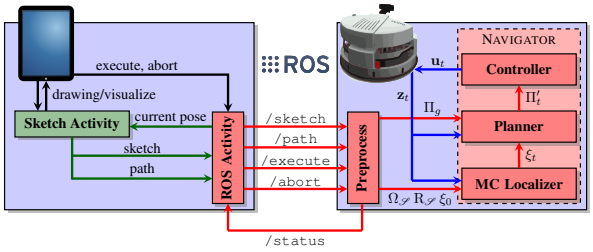

Fig. 2. System architecture showing the software components on the tablet and in the robot. As described in Sec III, ξ_t is the robot state. Π_g and Π'_t are respectively the global and local path, $\mathbf{u}_t$ is the control and $\mathbf{z}_t$ the measurements. $(\Omega_{\mathscr{S}}, \mathrm{R}_{\mathscr{S}})$ is the sketched map.

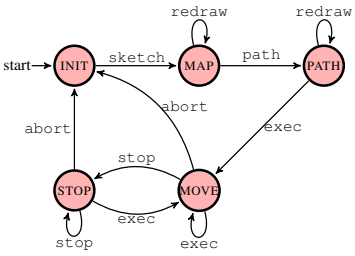

Fig. 3. A finite state machine depicting the tasks that a user can perform using the tablet interface.

uncertain landmarks and plan trajectories accordingly. Paths are therefore encoded as a set of waypoints output by a quadratic optimizer that accounts for the mutual position of the robot and estimated landmarks. Other approaches for matching the sketched scene with the real world have been suggested in [11] where Particle Swarm Optimization techniques are used to fit a hand-drawn sketch to an occupancy grid build using the current sensor data.

In our work, we present a theoretical framework for quantitatively interpreting a hand-drawn sketch of an indoor environment solely relying upon simple assumptions, namely topological consistency and small deformation. For this, we employ the Riemaniann manifold formalism and embed the sketch into a metric manifold whose metric tensor is unknown. Consequently, following [1] we estimate the metric together with the current robot pose using Monte Carlo Localization algorithm [22]. Once the conversion between current local metric in the sketch and the real world is known, a Dijkstra based planner is used to plan collision free paths in close proximity of the robot. This allows the robot to avoid unmapped obstacles as well as overcome minor inconsistencies in the sketch. In addition, we designed and implemented a tablet interface that allows a user to sketch a map of the environment and a path that the robot should follow for simple navigation and exploration tasks. A stack of planners autonomously handle small inconsistencies in the sketch as well as avoidance of unmapped obstacles, therefore the user is only required to provide a high level description of the environment.

The reminder of the paper is organized as follows. Section II outlines the design and core components of the tablet interface. In Section III we describe the navigation stack employed to perform the autonomous navigation tasks using hand-drawn maps. Finally in Section IV, we present the results from our experimental evaluation, both in terms of the autonomous navigation capability of the robot and from the usability perspective of the interface.

II. The Sketch Interface

The sketch interface was designed to run on a tablet or a mobile phone with a stylus or a touch interface. The overall system architecture shown in Fig. 2, was implemented using the Robot Operating System (ROS) framework and the interface components were implemented on the Android operating system. ROSJava, a Java based distribution of ROS was used in the Android application to publish and subscribe to topics to the ROS core running on the robot. The tablet and the robot communicate through WiFi.

The tasks that a user can perform using the interface can be represented as a finite state machine as shown in Fig. 3. The user is first presented with a canvas of size 2540 x 1252 pixels, in which he/she can sketch a map of the environment and draw polygons for obstacles. The sketched map is then sent to the robot when the user presses the *Send Sketch* button. The user then has the ability to draw the trajectory that the robot should take in the sketched environment. It is assumed that the user starts to draw the trajectory from the current position and orientation of the robot. There were no actual constraints set for the path to be drawn. The user can then send the sketched trajectory to the navigation system by pressing the *Send Path* button. The button is only activated and available to the user after the map is successfully sent.

The sketched map is encoded in the robot as a grid map, while the path is stored as a set of waypoints obtained by listening to touch events on the tablet. We interpret the initial position of the robot as the starting point of the path and set the initial orientation by estimating the direction of vector from the starting point to the next consecutive point beyond a preset threshold distance. This was done to avoid small squiggles in the beginning that affect the direction computation. A more detailed description on how the sketch is interpreted during the navigation tasks is given in Sec. III.

During both the map and path sketching, the user has the ability to redraw or erase parts of the sketch. This gives the user a very similar experience as drawing with a pencil and paper. The robot can then be instructed to navigate the sketched path by pressing the *Execute* button. The button is only activated and available to the user, after the navigational stacks on the robot have been initialized. This is notified to the interface using the `/status` command. He/She can also abort the mission at any point of time during the execution. A feedback message is displayed once the sketch and path are successfully sent and once the task is executing or is aborted.

III. Navigation in Hand-Drawn Maps

In order to interpret the hand-drawn sketch from the metrical perspective, we assume that a hand-drawn map $\mathscr{S} := (\Omega_{\mathscr{S}}, \mathrm{R}_{\mathscr{S}})$ is given as a rasterized image that describe a portion of plane $\Omega_{\mathscr{S}}$, with a own reference frame $R_{\mathscr{S}}$. Such

map describes qualitatively a real world indoor environment $\mathscr{W} := (\Omega_{\mathscr{W}}, \mathrm{R}_{\mathscr{W}})$ again encoded as a rasterized image. Under the assumption that the two images are topologically equivalent, we can further assume that there exists a diffeomorphism $\Phi : \Omega_{\mathscr{W}} \subset \mathbb{R}^2 \longrightarrow \Omega_{\mathscr{S}} \subset \mathbb{R}^2$ that transforms pixel by pixel the free space of the two images. As a consequence, we can describe the robot trajectory $(\mathbf{x}_t^{\mathscr{W}})_{t\geq 0} := ([x_t^{\mathscr{W}}, y_t^{\mathscr{W}}, \theta_t^{\mathscr{W}}])_{t\geq 0} \subset \mathrm{SE}(2)$ into the sketched world by applying the diffeomorphism Φ to the planar components of the robot poses, that is, $([\Phi(x_t^{\mathscr{W}}, y_t^{\mathscr{W}}), \theta_t^{\mathscr{W}}])_{t\geq 0}$. From trivial differentiation rules, it is apparent that the following integral relation holds:

$$\int_{\overrightarrow{\mathbf{x}}_0^{\mathscr{S}}}^{\overrightarrow{\mathbf{x}}_t^{\mathscr{S}}} d\overrightarrow{\mathbf{x}}^{\mathscr{S}} = \mathrm{T}_{\mathscr{S}\to\mathscr{W}} \int_{\overrightarrow{\mathbf{x}}_0^{\mathscr{W}}}^{\overrightarrow{\mathbf{x}}_t^{\mathscr{W}}} [\partial\Phi(x^{\mathscr{W}}, y^{\mathscr{W}})] d\overrightarrow{\mathbf{x}}^{\mathscr{W}}, \quad (1)$$

where $\partial\Phi : \Omega_{\mathscr{W}} \subset \mathbb{R}^2 \longrightarrow \mathbb{R}^{2\times 2}$ represents the Jacobian operator of the diffeomorphism Φ and the arrow notation the planar components of the pose. Thus, the motion of the robot in the real world can be translated into the sketch by means of a linear scaling operator or, more formally, the sketch can be taught as a differential (Riemaniann) manifold with metric tensor $\mathfrak{g}_{x,y} := \partial\Phi(x,y)^T \partial\Phi(x,y)$. Indeed, a chart for the sketch can be trivially defined via diffeomorphism by considering as new global coordinate system for the manifold the streamlines of Φ. As a consequence, owing to the fact that we can approximate the increment $d\mathbf{x}_t^{\mathscr{W}}$ by means of the odometry readings $\mathbf{u}_t$, namely $d\mathbf{x}_t^{\mathscr{W}} \approx \mathbf{x}_{t+1}^{\mathscr{W}} - \mathbf{x}_t^{\mathscr{W}} \approx \mathbf{x}_t^{\mathscr{W}} \oplus \mathbf{u}_t - \mathbf{x}_t^{\mathscr{W}}$, we can track the robot pose during the execution of a navigation task just exploiting the metric $\mathfrak{g}_{x,y}$ and the operator $\partial\Phi$.

Extending the idea presented in [1], in the rest of this section we describe how Monte Carlo Localization [22] can be adapt to track an approximation of the tensor metric on the sketch together with the current pose of the robot. A final section outlines the routine employed to track trajectories in the sketched map as well as how a local planning strategy can be used to avoid collisions with unmapped obstacles lying in the proximity of the robot.

A. Localization and Metric Estimation

To estimate the current pose of the robot while approximating the tensor metric $\mathfrak{g}_{x,y}$ we will henceforth assume that the local deformations are approximately shearing free. More precisely, we assume that

$$\mathfrak{g}_{x,y} \approx \begin{bmatrix} a(x,y)^2 & 0 \\ 0 & b(x,y)^2 \end{bmatrix}, \quad (2)$$

with $a(x,y), b(x,y) > 0$. To understand why this assumption is reasonable, observe that, up to a flip of the diagonal terms, from Eq. 2 and using the Singular Value Decomposition Theorem as well as restricting the case of sketches that are consistent in terms of orientation, the Jacobian of the diffeomorphism simplifies to

$$\partial\Phi = \begin{bmatrix} \cos\omega(x,y) & -\sin\omega(x,y) \\ \sin\omega(x,y) & \cos\omega(x,y) \end{bmatrix} \begin{bmatrix} a(x,y) & 0 \\ 0 & b(x,y) \end{bmatrix}. \quad (3)$$

Therefore, under the assumption that Eq. 2 holds, Eq. 3 can be read as the fact that the diffeomorphism applies a local distortion (stretch or compression) and a further local rotation.

Since we can assume that people are able to perceive orthogonality and parallelism of walls and it is a common experience to observe indoor environments mainly constituted of parallel and perpendicular walls, we can suppose that a sketch preserves a reasonable representation of parallel and orthogonal features. This hypothesis can be transferred into Eq. 3 by setting $\omega(x,y) \equiv \omega \in [0, 2\pi)$, that is, the local rotation is indeed a global one. Owing to this, the rotational component of the diffeomorphism can be therefore absorbed into the rotational term of the transformation $\mathrm{T}_{\mathscr{S}\to\mathscr{W}}$, provided that a suitable reference frame for the sketch has been choosen, say $\mathrm{R}_{\mathscr{S}}$. At a high level, this models the fact that the sketch could have been drawn with an arbitrary orientation, but still, it is inaccurate in terms of local stretching and compressions along the orthogonal coordinate systems of a reference frame $\mathrm{R}_{\mathscr{S}}$. As an example, the reader could think of a building with multiple rooms and a sketch which approximately preserve the directions of walls but the ratio of the sizes of the rooms are not accurate (as described in Fig. 4).

According to the above discussion, the Jacobian can be finally written as

$$\partial\Phi = R(\omega) \begin{bmatrix} a(x,y) & 0 \\ 0 & b(x,y) \end{bmatrix} =: [\mathrm{T}^{rot}_{\mathscr{S}\to\mathscr{W}}] S(x,y). \quad (4)$$

Since further details about how the reference frame $\mathrm{R}_{\mathscr{S}}$ is computed are not necessary at this point, we postpone the explanation to the next section.

In order to track both the position of the robot and the tensor metric during a navigation task, following the idea introduced in [1], we employ an extended version of the Monte Carlo Localization algorithm [22]. More precisely, we define the enhanced robot's state $\xi_t := (\mathbf{x}_t^{\mathscr{S}}, a_t, b_t)$ where $a_t, b_t > 0$ are the local scale with respect of the current robot position, namely $a(x_t^{\mathscr{W}}, y_t^{\mathscr{W}})$ and $b(x_t^{\mathscr{W}}, y_t^{\mathscr{W}})$. Accordingly, we apply the standard Bayes' filter, conditioning on the history of commands $\mathbf{u}_{1:t-1}$ and sensors' measurements $\mathbf{z}_{1:t}$, obtaining the following recursive update:

$$p(\xi_t \mid \mathbf{u}_{1:t-1}, \mathbf{z}_{1:t}) \propto p(\mathbf{z}_t \mid \xi_t) \cdot \int p(\xi_t \mid \xi_{t-1}, \mathbf{u}_{t-1})\, p(\xi_{t-1} \mid \mathbf{u}_{1:t-2}, \mathbf{z}_{1:t-1}) d\xi_{t-1}. \quad (5)$$

Standard Monte Carlo Localization approximates the state distribution using set of weighted samples, called particles. A first propagation step is performed and the particles are drawn according to a proposal distribution that encodes the evolution of the state with respect of the robot's commands and the surrounding environment. In the second step, the particles are "resampled" with *importance sampling* according to their weight, which is the likelihood of the current measurements if the observations would have been retrieved from the pose represented by the particle.

To select a *proposal distribution*, we assume that the scales a_t and b_t are independent one of the other as well as conditionally independent from the robot pose $\mathbf{x}_t^{\mathscr{S}}$, consequently

$$p(\xi_t \mid \xi_{t-1}, \mathbf{u}_t) \approx p(\mathbf{x}_{t-1}^{\mathscr{S}} \oplus \langle [\partial\Phi]_{t-1}, \mathbf{u}_{t-1} \rangle)\, p(a_t | a_{t-1})\, p(b_t | b_{t-1}) \quad (6)$$

being $[\partial\Phi]_{t-1}$ as in Eq. 4 with diagonal terms a_{t-1}, b_{t-1} and $\langle M, \cdot \rangle : \mathrm{SE}(2) \to \mathrm{SE}(2)$ the action of a matrix M on the planar components of $\mathrm{SE}(2)$. According to Eq. 6, the following model is a natural choice for describing the evolution of the state:

$$\begin{cases} \mathbf{x}_t^{\mathscr{S}} := \mathbf{x}_{t-1}^{\mathscr{S}} \oplus \langle [\partial\Phi]_{t-1}, \mathbf{u}_{t-1} + \boldsymbol{\varepsilon}_{t-1} \rangle, \\ a_t := a_{t-1}\gamma_{t-1}, \\ b_t := b_{t-1}\rho_{t-1}, \end{cases} \tag{7}$$

where $\boldsymbol{\varepsilon}_t \sim [\mathcal{N}_{0,\sigma_j}]_{j=1}^3$ is a random vector with independent normally distributed components (wrapped-normal for the orientation) and γ_t, ρ_t are independent multiplicative white noise, namely, with distribution $\Gamma_{\sigma_i^{-2},\sigma_i}$ $(i = 1, 2)$, the choice of the parameters ensures unitary mean and tunable variance.

As observation model for the likelihood function $p(\mathbf{z}_t \mid \xi_t)$ we extended the *likelihood fields model* for range finders described in [22]. Using the metrical conversion provided by $[\partial\Phi]_t$, we can convert the actual sensor readings in the sketch metric manifold. More precisely, let $\mathbf{z}_t = (z_{i,t})_{i=1}^N$ encoded as endpoints in the robot's reference frame. We set $\mathrm{T}_{\mathscr{S}\to\mathscr{R}}$ to be the transformation between $\mathrm{R}_{\mathscr{S}}$ and the robot's own coordinate system and we convert the readings endpoint in the sketch manifold by setting $z'_{i,t} := \mathrm{T}^{trans}_{\mathscr{S}\to\mathscr{R}} + S_t[\mathrm{T}^{rot}_{\mathscr{S}\to\mathscr{R}}]z_{i,t}$. Here S_t is the diagonal matrix defined in Eq. 4, computed with respect of the current scales a_t, b_t. Observe that the operation $S_t[\mathrm{T}^{rot}_{\mathscr{S}\to\mathscr{R}}]z_{i,t}$ is actually the projection of the readings on the tangent space of the sketch manifold. We then define the likelihood field for the raw sensor measurements as

$$p(\mathbf{z}_t \mid \xi_t) := \prod_{i=1}^{N} \mathcal{N}_{o'_{i,t},\sigma_i}(z'_{i,t}) \mathcal{L}_{\lambda,a_t}(a'_t) \mathcal{L}_{\nu,b_t}(b'_t), , \tag{8}$$

being $o'_{i,t}$ the closest obstacle to $z'_{i,t}$ in the sketch and $\mathcal{N}_{o'_{i,t},\sigma_i}$ Gaussian kernels. Although this model is extremely robust in standard metrical consistent maps [13], the presence of scaling factors can increase the change of "seeing through walls" effect, typical of endpoint models. To overcome this, we introduce another factor to bias the scaling factors. That is, we introduce a virtual measurement (a'_t, b'_t) for the scales obtained by solving the following least-square approximation: set $\mathbf{r}_i$ is the endpoint obtained ray casting along the direction of the endpoint $\mathbf{z}_{i,t}$ from the predicted robot's pose, we compute

$$\min_{A,B\in\mathbb{R}} \sum_{i=1}^{N} \left\{ [\pi_a(\mathbf{r}_i - Az_{i,t})]^2 + [\pi_b(\mathbf{r}_i - Bz_{i,t})^2] \right\}, \tag{9}$$

where and π_a and π_b are the projection along the axes centered in the robot position and aligned to the related scaling directions. Finally, we approximate the actual likelihood of the measurements as

$$p(\mathbf{z}_t \mid \xi_t) \approx \overline{p}(\mathbf{z}_t \mid \xi_t) \mathcal{L}_{\lambda,a_t}(a'_t) \mathcal{L}_{\nu,b_t}(b'_t), \tag{10}$$

where $\mathcal{L}_{\beta,s}$ is a Laplace distribution with mean s. The reason for choosing such distribution is to not suppress excessively the value lying far from the virtual scales since the ray casting procedure is unstable due to the inaccuracies of the sketch.

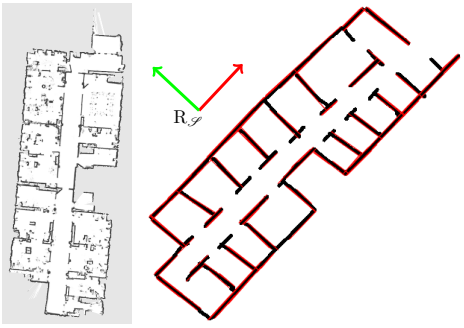

Fig. 4. Reference frame computed using the method described in Sec. III-B. On the left a SLAM image of an indoor building computed using the CARMEN framework [20]. On the right, the lab map sketched with our interface.

B. Estimating the Coordinate System on the Sketch

According to Sec. III-A, we assumed that the sketch contains only deformation along the directions of a suitable reference frame $\mathrm{R}_{\mathscr{S}}$. Applying the simple heuristic that walls in indoor environments are mainly orthogonal, we can select a coordinate system so that one of its axes is parallel to the most frequent direction of the walls and obstacles drawn in the sketch. To identify such direction we use an approach similar to the one suggested in [21]. More precisely, since a metrical consistent map $\Omega_{\mathscr{W}}$ can be thought as a chart for the sketch manifold and since the both $\Omega_{\mathscr{S}}$ and $\Omega_{\mathscr{W}}$ can be embedded in $\mathbb{R}^2$, we can set without ambiguity the world reference frame $\mathrm{R}_{\mathscr{W}}$ to be aligned to the pixel coordinates of the rasterized image $\Omega_{\mathscr{S}}$. Then we preprocess the image with the Canny's algorithm for edge detection [2]. Finally we run the Progressive Probabilistic Hough Transform [10] to obtain a set of direction $\{\theta_i\}_{i=1}^N \subset [0, 2\pi)$ with respect of $\mathrm{R}_{\mathscr{W}}$. In conclusion, to select the rotation angle for $\mathrm{T}_{\mathscr{S}\to\mathscr{W}}$ we run k-means on $\{\theta_i\}_{i=1}^N$ and set ω to be the mean of the biggest cluster. Such procedure identifies the direction of ω up to a rotation of π, however it is easy to see that this does not affect the behavior of the scaling factors. An example of the output is reported in Fig. 4.

C. Trajectory Tracking and Local Planning

In order to set up a navigation system that is able to track and execute a desired path in the sketch, we designed the robot's controller to have three different layers, namely:

- A *global planner* that stores a path drawn by a user.
- A *local planner* responsible for outputting collision free trajectories from the current robot position to the target waypoint on the global path. In this work, we use a Dijkstra planner on the local occupancy grid defined by the scaled readings $(z'_{i,t})_{i=1}^N$ (see Sec. III-A). A local planner that computes collision free trajectories is needed as the sketch should provide a high level description of the indoor environment without accounting for all the possible obstacles in the scene.
- A *trajectory tracker* that matches the current robot position with an approximate position on the desired path, with the aim of coordinating the two planners. It is apparent that, due to the presence of obstacles and inaccuracies

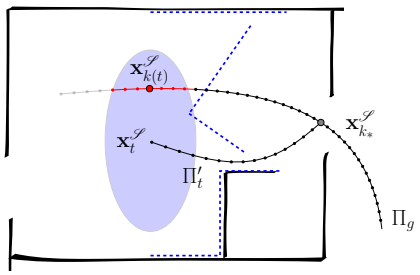

Fig. 5. Avoidance of unmapped obstacles in the sketch. In blue $W_L(\mathbf{x}^{\mathscr{S}}_t, r_L)$. The dashed blue line represents the scaled scan $(z'_{i,t})_{i=1}^N$.

in the sketch, only the local path is safe and consequently actuated by the robot. Thus the robot's trajectory can result in significant displacement from the desired global path, therefore a trajectory tracker is required.

In order to match the current position of the robot with a waypoint on the global path, we apply the strategy depicted in Fig. 5. That is, given a global path $\Pi_g := \{\mathbf{x}^{\mathscr{S}}_k\}_{k=1}^K$ and a current robot pose on the sketch $\mathbf{x}^{\mathscr{S}}_t$, we define the *local window* $W(\mathbf{x}^{\mathscr{S}}_t, r_L)$ to be the set of all poses $\mathbf{x}^{\mathscr{S}}$ so that $\|\mathbf{x}^{\mathscr{S}}_t - \mathbf{x}^{\mathscr{S}}\|_{\mathfrak{g}} < r_L$, where the norm applies only to the planar components. Here $r_L > 0$ is a parameter that can be expressed for example as length in pixels. Consequently, we select the subpath $\Pi'_t := W_L(\mathbf{x}^{\mathscr{S}}_t, r_L) \cap \Pi_g$ and consider the current position of the robot on Π_g to be the waypoint $\mathbf{x}^{\mathscr{S}}_{k(t)}$ that best approximates half of the arc length of Π'_t. In general Π'_t is not connected if a user has drawn a convoluted path. However, it is easy to discriminate which connected component of Π'_t should be chosen by following the ordering of the waypoints on Π_g and marking those that have already been visited.

To coordinate the two planners, we select a *lookahead window* $W_H(\mathbf{x}^{\mathscr{S}}_{k(t)}, r_H)$ depending on a parameter $r_H > 0$ as above and define the waypoint $\mathbf{x}^{\mathscr{S}}_{k_*} \in \Pi_g \cap W_H(\mathbf{x}^{\mathscr{S}}_{k(t)}, r_H)^{\complement}$ $(k(t) < k_*)$ to be the first waypoint in the path that lies outside the lookahead window. Finally, we plan a path in the sketch from $\mathbf{x}^{\mathscr{S}}_t$ to $\mathbf{x}^{\mathscr{S}}_{k_*}$ with respect of the scaled readings as discussed above.

IV. Experimental Evaluation

Experiments were carried out in an indoor environment built using temporary walls at the University of Freiburg. Obstacles were then placed at random locations inside the test environment. For carrying out the experiments, we used the Festo Robotino, an ominidirectional mobile platform equipped with a Hokuyo URG-04LX laser rangefinder. A picture of the test environment is shown in Fig. 1. The participants were first briefed about the task they had to perform and were shown the environment where the experiment was to be conducted. The participants did not have any technical knowledge on how the system worked or had seen the environment beforehand. As we envision the target users to be common people with different backgrounds, it was ensured that the participants were not experienced robot operators.

The task for them was to sketch a map of the environment and draw a path that they want the robot to follow in the sketched map. Most of the participants were not very familiar with drawing on a tablet so they were allowed sketch a few trials to get acquainted with the interface. The participants were not specifically instructed whether they should also draw the obstacles in the environment, this was intentional done in order to evaluate different scenarios.

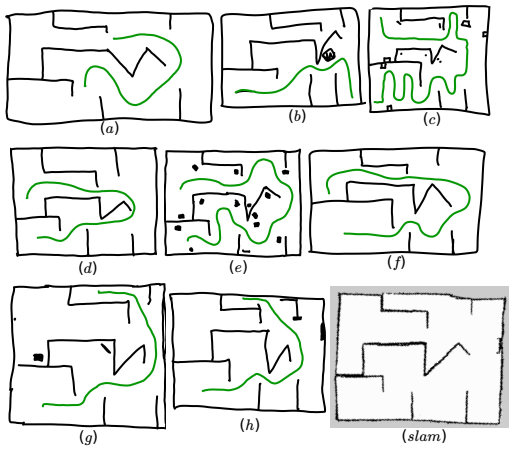

Fig. 6. Example sketches drawn by participants during the experiments. Some participants also sketch the obstacles. Bottom right, a map of the area obtained using Rao-Blackwallized SLAM.

There were a total of thirteen participants and they were split into two groups. The first group used the tablet in the landscape mode and the second group used the tablet in the portrait mode. We decided to conduct experiments using the tablet in different orientations because we noticed that users feel the urge to use the entire canvas to sketch the map, even if the proportions of the walls that they drew were very different from the real environment. Interestingly, this lead to different results in either cases. The results are discussed in the following sections. At the end of the experiment, the participants were asked to fill out a questionnaire and were also asked for suggestions to incorporate more intuitiveness into the interface. In order to maintain consistency in the evaluations, the environment was not altered in any way between each experiment cycle.

A. Usability Tests

There were significant variations in the sketches drawn by the participants. A few interesting examples are shown in Fig. 6. Some participants were concerned about drawing extremely straight lines for the walls but ignored drawing the obstacles (Fig 6(d), Fig 6(f)), whereas others were particular about drawing every obstacle in the environment (Fig. 6(c), Fig. 6(g)) but did not pay attention to the relative scales and positions of the walls (Fig. 6(a), Fig. 6(h)). Fig. 6(d) and Fig. 6(e) are some of the accurate sketches sufficiently depicting the environment. The time that the participants spent on drawing the maps varied from 27 seconds to over 6 minutes, while the average being 2.38 ± 1.42 minutes. Though there does not appear to be a pattern such as, the more time you spend on sketching, the higher is the success rate for the robot to complete the task, as each person pays attention to different parts of sketching and some spend considerable amount of

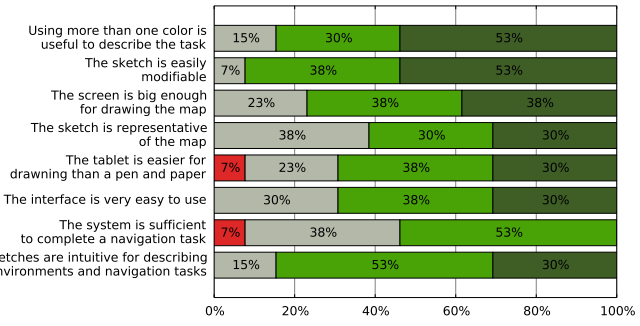

Fig. 7. Plot showing results from the post experiment survey. Strongly disagree, Disagree, Neutral, Agree, Strongly agree.

time erasing and redrawing the map. We also noticed that the attitude of the participants varied from one another, as some wanted to retry the experiment when the robot failed to navigate in their sketch, whereas some wanted to really push the navigation capabilities.

As mentioned in the description of the experiments, the participants performed experiments using the tablet in two different orientations. We found that the sketches drawn by the participants were more proportionally scaled, hence higher navigation success rate, when the tablet was used in the portrait orientation. As the screen real estate is smaller on the horizontal direction, it prevented them from drawing disproportionately rectangular sketches.

The questionnaire given to the participants was designed to get an insight on whether they felt at ease using the interface to complete the task at hand. We adopted the Likert scale [9] to rate the questions, with 5 (Strongly agree) being most satisfied and 1 (Strongly disagree) being the least. The survey revealed that using sketches to describe the environment was very intuitive for the participants, as they scored an average of 4.09. The users reported that having a small number of steps to perform in the interface to get the task done, the ability to edit the sketch and having multiple colors to sketch with, were all commendable.

Although only 30% of the participants strongly agree that the sketch is entirely representative of the environment and is easier to sketch than on paper, their comments revealed that this was because free-hand sketching on a tablet requires some practice and most participants had not sketched on a tablet before. This could be improved by providing the option of using predefined geometries for drawing. Almost no participant strongly agreed that the system is sufficient to complete the task, though 53% agreed. The users commented that this was because there was not enough feedback from the robot after the execute command is sent. Timely position updates and warnings or alerts can help provide more feedback to the user.

B. Navigational Autonomy

The experiments described above were also used to evaluate our navigation system. Overall, thirteen experiments were performed with a successful rate of 69.23% (9 successful runs of 13 sketches). The reliability of the entire system is dramatically affected by the quality of the sketch.

The parameters in the navigation stack were initially calibrated and were kept constant during the experiments. We tuned the parameters for odometry and sensor models using Monte Carlo Localization on metrically consistent map. The variances for the scales' model were chosen trading off the capability of adapting to the deformation of the sketch and the risk of increasing false detection. Similarly, the radius of the local lookahead window r_H affects the way the robot tracks the desired trajectory. If the radius is big, the robot is forced to track the locally optimal trajectory output by the Dijkstra planner. This results in considerable displacement from the drawn path if it is significantly suboptimal. However, if the parameter is chosen to be too small, the planner is not able to react quickly enough to unmapped obstacles and the safety of the navigation is severely affected.

All the failures occurred as a consequence of localization errors, in particular, it appears that the system is not robust to handle quick changes in the scales (the effect is visible in the right border of Fig. 6(g)). Moreover, we observed that participants drew sketches with different levels of details, but we observed that a navigation task was successful independent of the amount of clutter drawn in the sketch, for instance Fig. 6(b), Fig. 6(d),Fig. 6(e) were successful, while Fig. 6(c) was not.

V. Conclusions

In this paper we addressed the problem of equipping a non-expert user with an interactive tool using which spatial information about the environment can be communicated to the robot. To accomplish this, we designed and implemented a tablet interface that allows a user to sketch a map of an indoor environment and specify a desired trajectory that the robot should follow. We presented a theoretical framework that enables the robot to localize itself with respect to the hand-drawn map, which is achieved by tracking the pose of the robot together with a local metric of the sketch. We further use this metrical description to convert the sensor's readings into the sketched environment and use these virtual measurements to perform avoidance of unmapped obstacles as well as to overcome small inconsistencies in the drawing.

We performed a usability study of our interface to determine how practical it is to sketch a map of the environment that sufficiently describes the real-world, in order to successfully carry out a navigation task. We found that each user has a very different style and focus while sketching a map and the system has to be robust to all the variations of the sketch. Nevertheless, we have shown that even in a cluttered environment, a minimal representation of the scene as a sketch is adequate for successfully navigating it.

Acknowledgements

This work has been supported by the European Commission under contract numbers FP7–610532–SQUIRREL, FP7–610603–EUROPA2, Horizon 2020–645403–RobDREAM.

References

[1] B. Behzadian, P. Agarwal, W. Burgard, and G. D. Tipaldi. Monte Carlo localization in hand-drawn maps. 2015. arXiv:1504.00522v1.

[2] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.

[3] G. Chronis and M. Skubic. Sketch-based navigation for mobile robots. In *Proc. of the 12th IEEE International Conference on Fuzzy Systems, 2003. FUZZ'03*, volume 1, pages 284–289. IEEE, 2003.

[4] G. Grisetti, G.D. Tipaldi, C. Stachniss, W. Burgard, and D. Nardi. Fast and accurate SLAM with rao-blackwellized particle filters. *Robotics and Autonomous Systems*, 55(1):30–38, 2007. URL http://ais.informatik.uni-freiburg.de/~tipaldi/papers/grisettiRAS07.pdf.

[5] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010. doi: 10.1109/MITS.2010.939925. URL http://ais.informatik.uni-freiburg.de/publications/papers/grisetti10titsmag.pdf.

[6] S. Hemachandra, M. R. Walter, S. Tellex, and S. Teller. Learning spatial-semantic representations from natural language descriptions and scene classifications. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2623–2630. IEEE, 2014.

[7] K. Kawamura, A.B. Koku, D. M. Wilkes, R. A. Peters, and A. Sekmen. Toward egocentric navigation. *International Journal of Robotics and Automation*, 17(4):135–145, 2002.

[8] S. Koenig and R. G. Simmons. Passive distance learning for robot navigation. In *ICML*, pages 266–274, 1996.

[9] Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[10] J. Matas, C. Galambos, and J. Kittler. Robust detection of lines using the Progressive Probabilistic Hough Transform. *Computer Vision and Image Understanding*, 78(1): 119–137, 2000.

[11] G. Parekh, M. Skubic, O. Sjahputera, and J. M. Keller. Scene matching between a map and a hand drawn sketch using spatial relations. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4007–4012. IEEE, 2007.

[12] A. Pronobis and P Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3515–3522. IEEE, 2012.

[13] J. Roewekaemper, C. Sprunk, G.D. Tipaldi, C. Stachniss, P. Pfaff, and W. Burgard. On the position accuracy of mobile robot localization based on particle filters combined with scan matching. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3158–3164, Villamoura, Protugal, 2012. URL http://ais.informatik.uni-freiburg.de/publications/papers/roewekaemper12iros.pdf.

[14] D. C. Shah and M. E. Campbell. A robust qualitative planner for mobile robot navigation using human-provided maps. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2580–2585. IEEE, 2011.

[15] M. Skubic, C. Bailey, and G. Chronis. A Sketch interface for mobile robots. In *Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC)*, volume 1, pages 919–924. IEEE, 2003.

[16] M. Skubic, S. Blisard, C. Bailey, J. A Adams, and P. Matsakis. Qualitative analysis of sketched route maps: Translating a sketch into linguistic descriptions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(2):1275–1282, 2004.

[17] M. Skubic, D. Anderson, S. Blisard, D. Perzanowski, W. Adams, J. G. Trafton, and A. C. Schultz. Using a sketch pad interface for interacting with a robot team. In *Proc. of AAAI*, volume 5, pages 1739–1740, 2005.

[18] M. Skubic, D. Anderson, S. Blisard, D. Perzanowski, and A. Schultz. Using a hand-drawn sketch to control a team of robots. *Autonomous Robots*, 22:399–410, 2007.

[19] Marjorie Skubic, Pascal Matsakis, Benjamin Forrester, and George Chronis. Extracting navigation states from a hand-drawn map. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 259–264. IEEE, 2001.

[20] C. Stachniss. URL http://www2.informatik.uni-freiburg.de/~stachnis/datasets/datasets/fr079/fr079-complete.gfs.png.

[21] S Thrun and D Dolgov. Detection of principal directions in unknown environments for autonomous navigation. *Robotics: Science and Systems*, pages 73–79, 2009.

[22] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[23] M. R. Walter, S. Hemachandra, B. Homberg, S. Tellex, and S. Teller. Learning semantic maps from natural language descriptions. Robotics: Science and Systems, 2013.

[24] H. Zender, O. Martiínez Mozos, P. Jensfelt, G-J.M. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502, 2008.