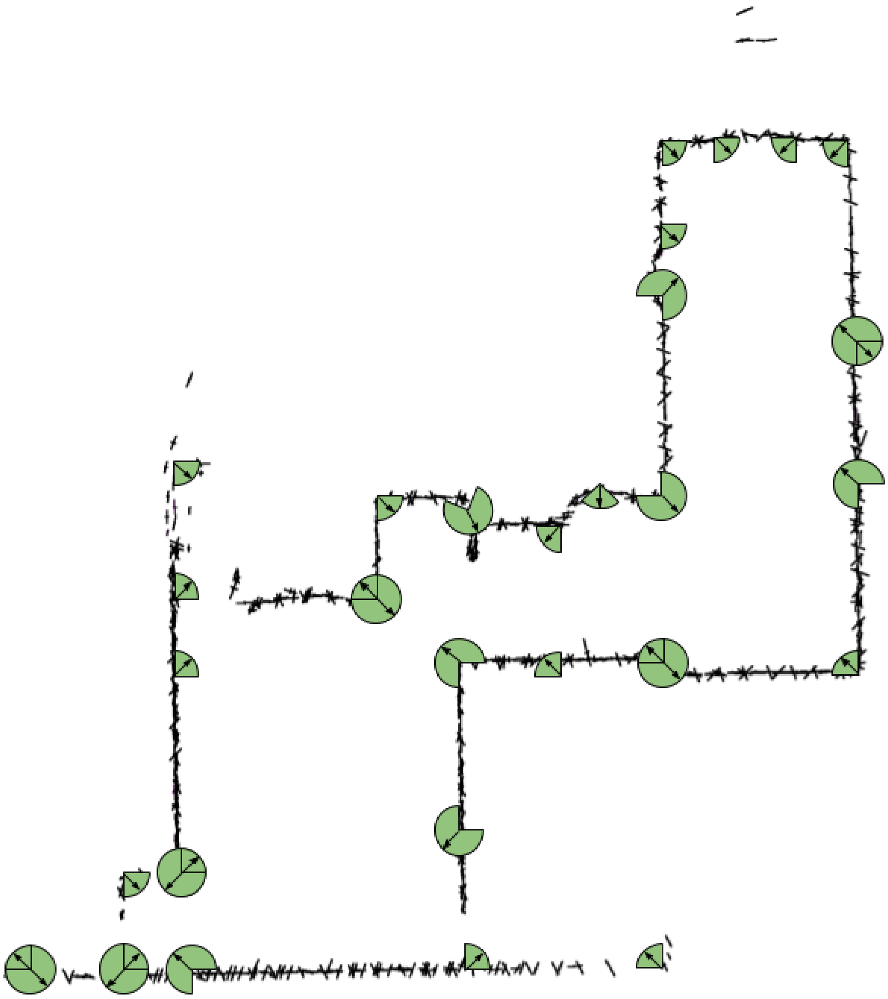

Figure 5. NDT map with all corners, and their orientations and widths. Some corners are detected along walls either due to noise in the measurements or details on the walls such as door frames.

4.2. Feature Extraction and Robot Localization in the Layout Map

We aim at merging the information seen by the robot and its sensors with the information given by a rough layout map, and focus on using emergency maps. Interpreting emergency map symbols is outside the scope of this work (one should mention the work of de las Heras et al. [25] and Ahmed et al. [26], who interpreted the different elements of a layout map and whose work could be used to solve that preprocessing step). In our work, we converted the emergency maps by hand to a layout map representing only the walls in the environment. Since corners and walls are common features, we extracted them in the layout map using a line follower algorithm [1]; each wall is approximated by a straight line between its two corners. The covariance associated to each wall is determined as in our previous work [1]: we use as the main eigenvector for the covariance $n\%$ of the main axis of the wall and we assign a small length of 0.05 m to the second perpendicular eigenvector. Hence, the walls 垂直的 in the layout map will easily extend or shrink but will be hard to rotate. The amount of deformation allowed on the layout map depends on the user chosen parameter n. In our work, we use $n = 10\%$.

We estimate the position of the robot relative to the layout map by performing Monte-Carlo localization (MCL) on each incoming scan. The MCL algorithm is based on the method presented by Saarinen et al. [4]. However, we extend it to take into account local uncertainties of the layout map, such as local scaling errors, missing elements of the environment, or errors in representation or perspective. The MCL filter populates the space with a certain number of possible poses, or particles.

如何进行定位?
稀疏和定位

For each particle, incoming laser scans are scored against the layout map, and, as more laser scans are received, particles that fit the incoming data better are given a higher probability, and consequently have a higher chance of being resampled for the next iteration. In time, the particles for which the incoming data fits the layout map best will have the highest probability. Hence, the population of particles will tend to focus around the true robot pose.

particle 滤波?

找到最适合的 scan

The algorithm presented by Saarinen et al. [4] assumes a correct model map. Thus, for each cell of the input scan seen from a given particle, the score is the likelihood that two NDT cells are the same, calculated using the l^2 norm *only if* a cell with a Gaussian is present at the same place in the model map. Cells further away are ignored and the l^2 norm is used to reduce the effect of outliers in otherwise correctly matching scans and model maps. On the other hand, the l^2 norm is not adapted for maps with large uncertainties: it is a function of the distance between the mean of the NDT cells and the product of their Gaussians:

$$l^2 = \vec{\mu}(\Sigma_1 \Sigma_2)^{-1} \vec{\mu}^\intercal \tag{1}$$

with Σ_1 and Σ_2 the covariances of both NDT cells, and $\vec{\mu}$ the vector between their centers. Hence, if the product of the Gaussians is small, a very small distance between the NDT cells' means is enough to get a large l^2 norm value. In rough model maps, uncertainty in the representation would be enough to get very high l^2 norm, lowering the score of correct particles. Hence, the pose and covariance returned by the MCL filter would ignore all scale or detail inaccuracies in the layout map.

However, the sensor model used by the MCL filter needs to take into consideration the uncertainties in representation of the layout map. Instead of the standard NDT-MCL sensor model, we use the Euclidean distance d_E between the means of two NDT cells, and calculate the final score as:

$$\text{score} = 0.1 + 0.9 * e^{-s * d_\text{E}} \tag{2}$$

where s is scaling the likelihood that both cells are the same so that every distance from 0 up to the neighbor size *n_size* is returning a score between 0.1 and 1. Hence, s is expressed such as:

$$s = \frac{4}{n * r} \tag{3}$$

with n a user chosen parameter determining the size of the neighborhood considered given in meters and r the resolution of the map. For every Gaussian in the input scan, if no cell with a Gaussian is present at the same position in the layout map, we consider a neighborhood of cells of size n in the layout map. In that case, the final score is the mean score of all neighboring cells. Since cells further away have a lower score than a cell nearby, one close cell in the layout map will have more influence than the mean score of a neighborhood of cells.

To initialize the MCL filter, we assume that the robot has an approximation of the scale of the emergency map and knows its approximate starting point in the layout map. Those are not strong assumptions as, in operation, the first responders are usually aware of the position from which the robot starts its exploration in the layout map. Furthermore, emergency maps are semi-metric maps and thus provide a scale between the map and the environment. The space around the robot start pose is populated with particles using a Gaussian probabilistic distribution for the position, while the angle of each particle is kept the same as that of the starting pose. Hence, the first robot pose added in the ACG corresponds to the approximate starting pose of the robot in the layout map. The accuracy of the starting pose should be good enough for the MCL filter to converge to the correct pose in the layout map, i.e. the population of particles should focus around the true robot pose after a couple of particle resamplings. Hence, the accuracy of the starting point in the layout map depends on both the initial covariance of the MCL filter and on the layout map's ambiguity in representation. In our work, the robot pose in the emergency map was given by the user through a graphical interface displaying the emergency map (see Figure 15). Furthermore, the user could use the interface to approximate the scale of the layout map by clicking on a point in the layout map coordinate frame and then clicking on