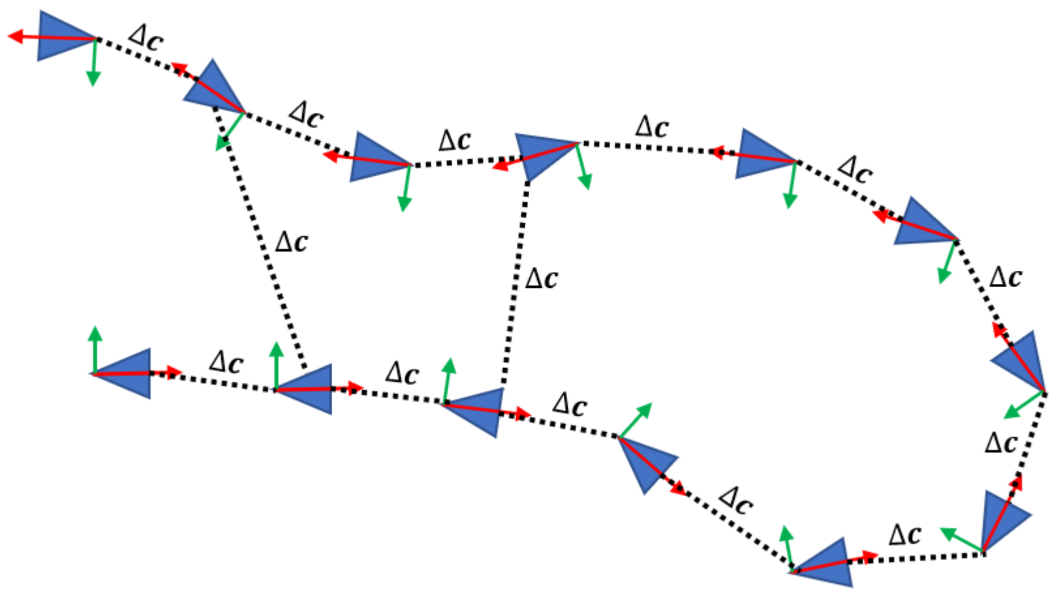
Vision Aided Navigation - Exercise 5

In **exercise 4** we summarized the information from the Bundle optimizations in a **Pose Graph**.

We use the pose graph as a concise description of the trajectory. This will enable us to recognize that the current location of the vehicle is (possibly) similar to some past location and initiate a search for the exact relative pose between two frames.

When we find such connection to a past frame we'll use it to add a **Loop Closure** constraint to the pose graph, thus greatly reducing the drift of the trajectory estimation.



Pose Graph: Relative poses Δc between consecutive poses and distant poses (loop closure)

5

For each key frame c_n in the pose graph loop over previous frames $c_i,\ i < n$, and perform steps 5.1-5.4:

5.1 Detect Loop Closure Candidates

a. Relative Covariance

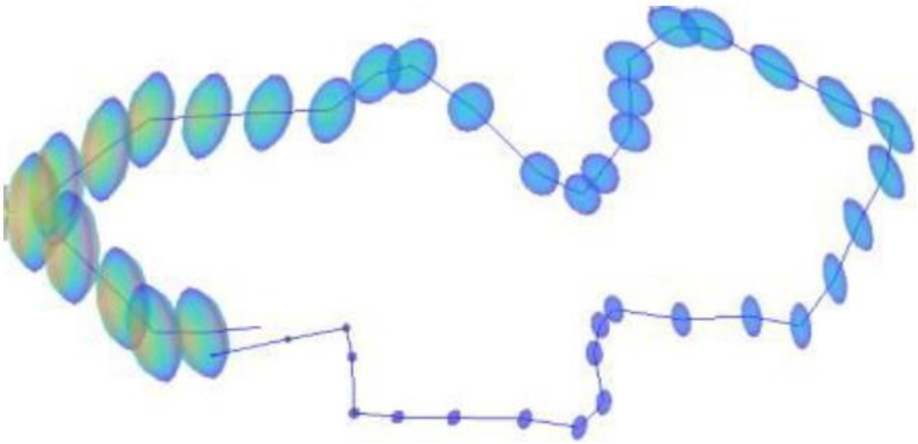
find the shortest path from c_n to c_i and compose the incremental covariances along the path to get an estimate of the relative covariance $\Sigma_{n|i}$.

- What are reasonable choices for the edge weights? i.e. in what way is the chosen path the shortest?
  What did you use as edge weights?
- How did you implement the shortest path algorithm?

b. Detect Possible Candidates

Choose the most likely candidate to be close to the pose c_n by applying a Mahalanobis distance test $\Delta c_{ni}^{T} \Sigma_{n|i}^{-1} \Delta c_{ni}$ with c_{ni} the relative pose between c_n and c_i.

Choose a threshold to determine if the candidate advances to the next (expensive) stage.



Relative Covariance

Note that the last frame uncertainty threshold includes the 1st frame location

c. Optional - Use DNN

As an alternative to **a,b** use DNN of your choice to extract a descriptor for the left image of c_n and match to all previous descriptors. Choose the best match if it is below a chosen threshold.

- What DNN did you choose? Why?
- What did you use as descriptor? How long did it take to compute?

5.2 Consensus Matching

Perform consensus match between the two candidate frames. (See exercise 2)

Set a threshold for the number of inliers that indicates a successful match. Note that this is typically a more difficult match than that of two consecutive frames.

- What was your chosen threshold?

Two similar frames from different times with a successful consensus match (inlier in cyan)

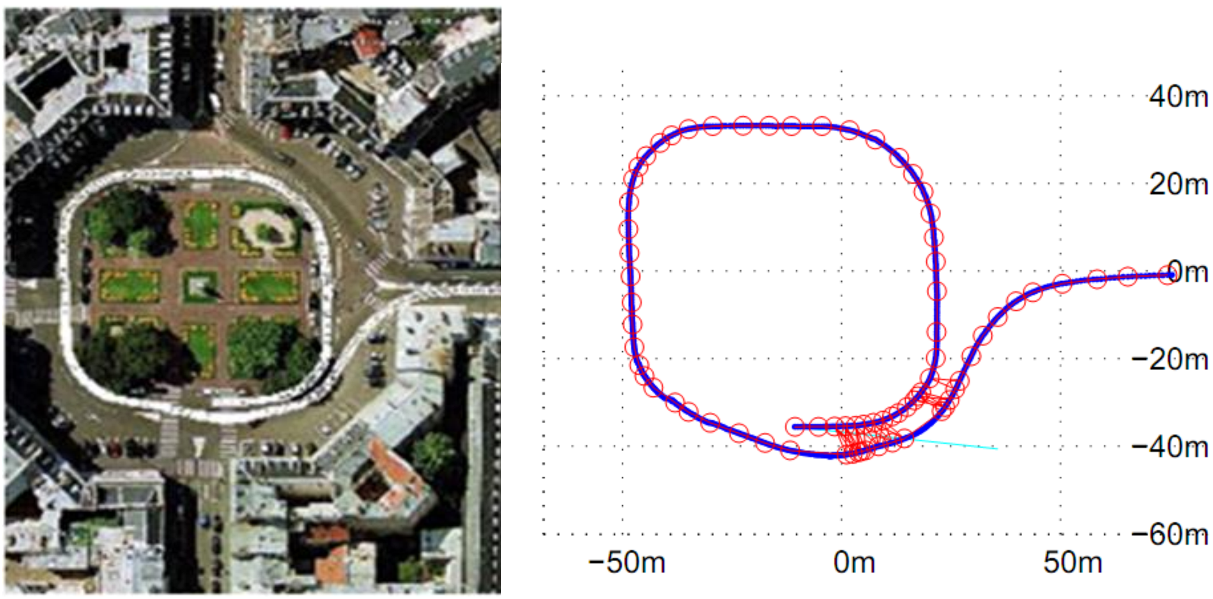
5.3 Relative Pose Estimation

Using the inlier matches perform a small Bundle optimization to extract the relative pose of the two frames as well as its covariance.

- What is a reasonable initial solution for the Bundle?
- How did you extract the appropriate covariance?

5.4 Update the Pose Graph

Add the resulting synthetic measurement to the pose graph and optimize it to update the trajectory estimate.



Trajectory around a square. Estimation on the right – keyframes marked in circles, loop closure edges in red

5.5

- How many successful loop closures were detected?
- Plot the match results of a single successful consensus match of your choice. (For the left images, inliers and outliers in different colors)
- Choose 5 versions of the pose graph along the process and plot them (including location covariance).
  Explain at what points you chose to plot the graph.
- Plot a graph of the absolute location error for the whole pose graph both with and without loop closures.
- Plot a graph of the location uncertainty size for the whole pose graph both with and without loop closures. (What measure of uncertainty size did you choose?)

GTSAM

- *pose_prior = gtsam.PriorFactorPose3(key, pose, uncertainty)*
- Factor error for particular values:
  pose_prior.error(gtsam.Values())
- *gtsam.noiseModel.Gaussian.Covariance(S)*
- *relative_pose = pose_c0.between(pose_c1)*
- *gtsam.BetweenFactorPose3(c0, c1, relative_pose, noiseCov)*