APPENDIX

A. Proof of the Derivative of the Rotation Angle Function

Lemma 8, copied below, gives the derivative of the rotation angle function along an edge of a rotation cube $\mathcal{C}_r$. Here we present the full derivation of this result.

Lemma 8. *(Derivative of the rotation angle function) Given a unit 3D bearing vector* $\mathbf{f}$ *and a rotation cube* $\mathcal{C}_r$ *centred at* $\mathbf{r}_0$ *with vertices* $\{\mathbf{r}_i\}_{i\in[1,8]}$*, then the derivative of the rotation angle function*

$$A_{ij}(\lambda) = \arccos\big((\mathbf{R}_{\mathbf{r}_0}^{-1}\mathbf{f}) \cdot (\mathbf{R}_{\mathbf{r}_{ij}(\lambda)}^{-1}\mathbf{f})\big) \tag{A.1}$$

with respect to λ*, for an edge parametrisation of* $\mathbf{r}_{ij}(\lambda) = \mathbf{r}_i + \lambda(\mathbf{r}_j - \mathbf{r}_i)$ *with* $\lambda \in [0,1]$*, is given by*

$$\frac{dA_{ij}}{d\lambda} = \frac{\mathbf{f}^\intercal \mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}_{ij}(\lambda)}^\intercal[\mathbf{f}]_\times\Big(\mathbf{r}_{ij}(\lambda)\mathbf{r}_{ij}(\lambda)^\intercal - (\mathbf{R}_{\mathbf{r}_{ij}(\lambda)} - I)[\mathbf{r}_{ij}(\lambda)]_\times\Big)\Big(\mathbf{r}_i - \mathbf{r}_j\Big)}{\|\mathbf{r}_{ij}(\lambda)\|^2\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}_{ij}(\lambda)}^\intercal\mathbf{f})^2}}. \tag{A.2}$$

Proof. Equation (A.2) can be derived as follows:

$$\begin{aligned}
\mathrm{d}A &= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathrm{d}\big((\mathbf{R}_{\mathbf{r}_0}^{-1}\mathbf{f})\cdot(\mathbf{R}_{\mathbf{r}(\lambda)}^{-1}\mathbf{f})\big) && \text{(A.3)}\\
&= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathrm{d}\big(\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f}\big) && \text{(A.4)}\\
&= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathrm{d}\big(\mathbf{R}_{-\mathbf{r}(\lambda)}\mathbf{f}\big) && \text{(A.5)}\\
&= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\left(-\mathbf{R}_{-\mathbf{r}(\lambda)}[\mathbf{f}]_\times\frac{(-\mathbf{r}(\lambda))(-\mathbf{r}(\lambda))^\intercal + (\mathbf{R}_{-\mathbf{r}(\lambda)}^\intercal - I)[-\mathbf{r}(\lambda)]_\times}{\|-\mathbf{r}(\lambda)\|^2}\right)\mathrm{d}(-\mathbf{r}(\lambda)) && \text{(A.6)}\\
&= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\left(-\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal[\mathbf{f}]_\times\frac{\mathbf{r}(\lambda)\mathbf{r}(\lambda)^\intercal - (\mathbf{R}_{\mathbf{r}(\lambda)} - I)[\mathbf{r}(\lambda)]_\times}{\|\mathbf{r}(\lambda)\|^2}\right)\mathrm{d}(-\mathbf{r}(\lambda)) && \text{(A.7)}\\
&= \frac{-1}{\sqrt{1-(\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal\mathbf{f})^2}}\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\left(\mathbf{R}_{\mathbf{r}(\lambda)}^\intercal[\mathbf{f}]_\times\frac{\mathbf{r}(\lambda)\mathbf{r}(\lambda)^\intercal - (\mathbf{R}_{\mathbf{r}(\lambda)} - I)[\mathbf{r}(\lambda)]_\times}{\|\mathbf{r}(\lambda)\|^2}\right)(\mathbf{r}_j - \mathbf{r}_i)\mathrm{d}\lambda && \text{(A.8)}
\end{aligned}$$

where (A.6) uses Result 1 from [2] and the derivative follows from the differential. □

Since the derivative is computationally expensive to calculate, we only evaluate it at the vertices. In addition, we only require the sign of the result, which simplifies the equation. Corollary A.1 presents the relevant results.

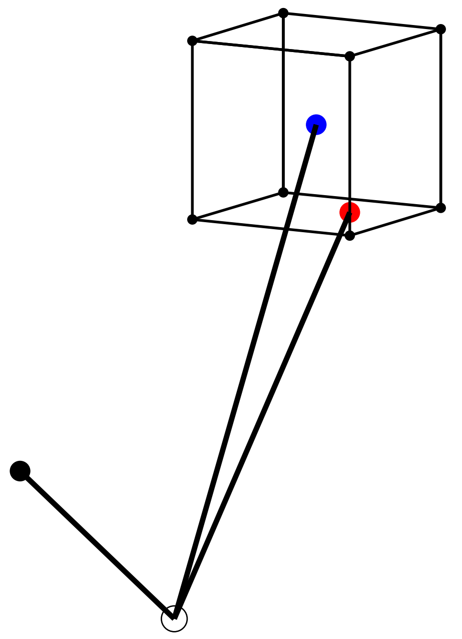
Corollary A.1. *(Sign of the derivative of the rotation angle function at the vertices) Given a unit 3D bearing vector* $\mathbf{f}$ *and a rotation cube* $\mathcal{C}_r$ *centred at* $\mathbf{r}_0$ *with vertices* $\{\mathbf{r}_i\}_{i\in[1,8]}$*, then*

$$\operatorname{sgn}\left.\frac{dA}{d\lambda}\right|_{\lambda=0} = \operatorname{sgn} -\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}_i}^\intercal[\mathbf{f}]_\times\Big(\mathbf{r}_i\mathbf{r}_i^\intercal - (\mathbf{R}_{\mathbf{r}_i} - I)[\mathbf{r}_i]_\times\Big)\Big(\mathbf{r}_j - \mathbf{r}_i\Big) \tag{A.9}$$

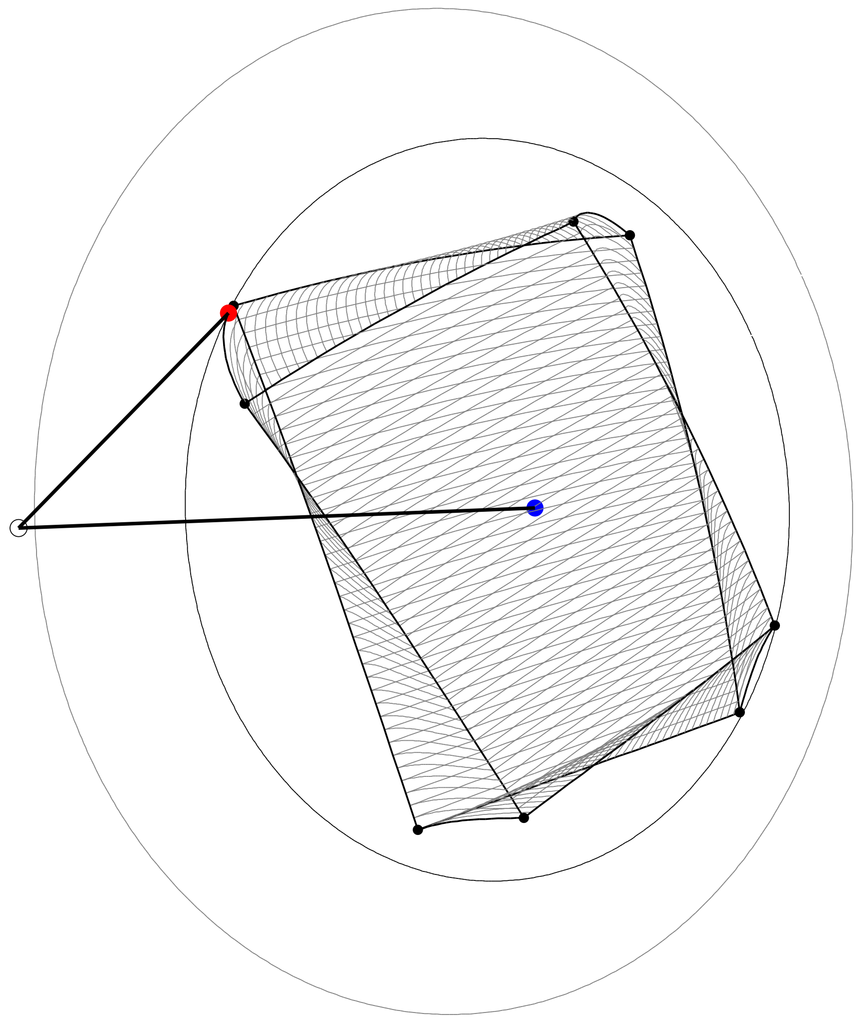
and

$$\operatorname{sgn}\left.\frac{dA}{d\lambda}\right|_{\lambda=1} = \operatorname{sgn} -\mathbf{f}^\intercal\mathbf{R}_{\mathbf{r}_0}\mathbf{R}_{\mathbf{r}_j}^\intercal[\mathbf{f}]_\times\Big(\mathbf{r}_j\mathbf{r}_j^\intercal - (\mathbf{R}_{\mathbf{r}_j} - I)[\mathbf{r}_j]_\times\Big)\Big(\mathbf{r}_j - \mathbf{r}_i\Big). \tag{A.10}$$

When (A.9) is positive and (A.10) is negative, the angle maximiser occurs on the edge and not at a vertex. An example of the small fraction of cases where this occurs is shown in Fig. A.1. The figure also demonstrates the validity of the assumption that the maximum always occurs on the cube skeleton (edges and vertices), not the faces. It can be observed that rotation vectors on the cube faces are not projected beyond the convex hull of the projection of the edges for a given point.



(a) Rotation cube in angle-axis space with centre $\mathbf{r}_0$ (blue dot), projected angle maximiser $\mathbf{r}^*$ (red dot), origin (black circle) and unrotated 3D point $\mathbf{p}$ (black dot). Cube edges and vertices are shown as thin black lines and small black dots respectively.



(b) Rotation of 3D point $\mathbf{p}$ by angle-axis vectors on the surface of the cube with centre-rotated point $\mathbf{R}_{\mathbf{r}_0}\mathbf{p}$ (blue dot), angle maximiser $\mathbf{R}_{\mathbf{r}^*}\mathbf{p}$ (red dot) and origin (black circle). 40 equally-spaced lines across each face are plotted in grey. All points and lines, other than the origin and lines to the origin, lie on the surface of a sphere with radius $\|\mathbf{p}\|$. Cube edges and vertices are shown as thin black lines and small black dots respectively. The weak rotation uncertainty angle ψ_r^w corresponds to the aperture angle of the cone formed by the origin and the grey circle. Our rotation uncertainty angle ψ_r corresponds to the aperture angle of the cone formed by the origin and the black circle.

Figure A.1. A random rotation cube and the rotation of a random 3D point by all angle-axis vectors on the surface of that cube. Observe that the rotation vector that maximises the angle $\angle(\mathbf{R}_\mathbf{r}\mathbf{p}, \mathbf{R}_{\mathbf{r}_0}\mathbf{p})$ lies on a cube edge. Also observe that rotation vectors on the face of the cube (grey lines in the projection) do not rotate the point beyond the convex hull of the point rotated by the edges. Best viewed in colour.

B. Precomputing Angles on the Sphere

To reduce the time complexity of the bound calculations, the angle between the translated 3D points $\mathbf{p} - \mathbf{t}_0$ and any location on the unit sphere may be precomputed. Thus, for a fixed translation, the angle between *any* rotated bearing vector and its rotationally-closest 3D point may be precomputed. This is the analogue in S^2 of a distance transform in $\mathbb{R}^n$, in that the surface of the sphere is discretised and a look-up table constructed. It exploits the nested structure of the algorithm, since many bounds for different rotations are calculated for a single translation. By using this precomputation, the $\max$ operations in (42)–(45) are reduced from $\mathcal{O}(M)$ to $\mathcal{O}(1)$.

The look-up table was constructed by linearly projecting the sphere onto an enclosing cube whose faces were partitioned with quad-trees. We subdivided the sphere into 98304 regions, such that the maximum angle between a point and its nearest cell centre was $0.6°$. While the linear projection produces non-uniform cell sizes, it facilitates the rapid conversion from a unit vector to a location in the data structure. Since this data structure relaxes θ by up to $0.6°$, it is not optimal with respect to θ. However, it may be useful for problems with a large number of 3D points M. For the purposes of evaluating the algorithm, this feature was not used in the experiments.

C. Time Complexity Analysis

Explicitly including the tolerance η in the bound formulae makes it possible to derive a bound on the worst-case search tree depth and thereby obtain the time complexity of the algorithm. In terms of the size of the input, the GOPAC algorithm is $O(MN)$, or $O(N)$ if angle precomputation is used, where M is the number of 3D points and N is the number of bearing vectors. However, the notation conceals a very large constant. Including the constant factors that can be selected by the user yields $O(\rho_{t_0}^3 \zeta^{-3} \eta^{-6} MN)$, where ρ_{t_0} is the half space diagonal of the initial translation cuboids, that is one-quarter the space diagonal of the translation domain, and ζ and η are small previously-defined constants set by the user.

Calculating the upper and lower bounds involves a summation over $\mathcal{F}$ and a maximisation over $\mathcal{P}$, therefore the complexity is $O(MN)$. If angle precomputation is used, the maximisation becomes a constant-time lookup leading to a bound complexity of $O(N)$. However, it is as of yet unclear how the number of iterations (explored subcuboids) depends on the inputs. The central finding is that branch-and-bound is exponential in the worst-case tree search depth D, but D is logarithmic in η^{-1}. Therefore the complexity of BB is polynomial in η^{-1}, where η is the angle tolerance. Rotation and translation search will be treated separately before being combined into an analysis of nested rotation and translation search.

Theorem C.1. *(Rotation Search Depth and Complexity) Let $\rho_{r_0} = \sqrt{3}\delta_{r_0} = \sqrt{3}\pi/2$ be the half space diagonal of the initial rotation subcube $\mathcal{C}_{r_0}$. Then*

$$D_r = \max\left\{\left\lceil \log_2 \frac{\rho_{r_0}}{\eta} \right\rceil, 0\right\} \tag{C.1}$$

is an upper bound on the worst-case rotation tree search depth for an uncertainty angle tolerance η and $O(\eta^{-3})$ is the time complexity of rotation BB search.

Proof. Rotation BB converges when $\underline{\nu}_r \geqslant \bar{\nu}_r$. For any $\psi_t'(\mathbf{p}, \mathcal{C}_t)$ in (42) and (43), $\underline{\nu}_r \geqslant \bar{\nu}_r$ when $\psi_r'(\mathbf{f}, \mathcal{C}_r) \leqslant 0$ or equivalently $\psi_r(\mathbf{f}, \mathcal{C}_r) \leqslant \eta$ for all $\mathbf{f} \in \mathcal{F}$. Now,

$$\psi_r(\mathbf{f}, \mathcal{C}_r) = \min\left\{\max_{\mathbf{r}\in\mathcal{S}_r} \angle(\mathbf{R_r f}, \mathbf{R}_{\mathbf{r}_0}\mathbf{f}), \pi\right\} \tag{C.2}$$

$$= \min\left\{\angle(\mathbf{R}_{\mathbf{r}^*}\mathbf{f}, \mathbf{R}_{\mathbf{r}_0}\mathbf{f}), \pi\right\} \tag{C.3}$$

$$\leqslant \min\left\{\sqrt{3}\delta_r, \pi\right\} \tag{C.4}$$

$$\leqslant \rho_r \tag{C.5}$$

where (C.3) replaces the maximisation with the $\arg\max$ rotation $\mathbf{r}^*$, (C.4) follows from Lemma 2 and ρ_r is the half space diagonal of the rotation subcube $\mathcal{C}_r$. At rotation search tree depth D_r, the half space diagonal is given by

$$\rho_{r_{D_r}} = \frac{1}{2}\rho_{r_{D_r - 1}} = \frac{1}{2^{D_r}}\rho_{r_0}. \tag{C.6}$$

Substituting into (C.5) gives

$$\psi_r(\mathbf{f}, \mathcal{C}_{r_{D_r}}) \leqslant \rho_{r_{D_r}} = 2^{-D_r}\rho_{r_0}. \tag{C.7}$$

To find the worst-case rotation search tree depth, the constraint $\psi_r(\mathbf{f}, \mathcal{C}_r) \leqslant \eta$ is applied:

$$\psi_r(\mathbf{f}, \mathcal{C}_{r_{D_r}}) \leqslant 2^{-D_r}\rho_{r_0} \leqslant \eta. \tag{C.8}$$

Taking the logarithm of both sides yields

$$D_r \geqslant \log_2 \frac{\rho_{r_0}}{\eta}. \tag{C.9}$$

Equation (C.1) follows from the requirement that D_r be a non-negative integer. Now, rotation BB will have examined at most

$$N_r = 8(1 + 8 + 8^2 + \cdots + 8^{D_r}) = 8\frac{8^{D_r+1} - 1}{8 - 1} = \frac{8}{7}\left((2^{D_r+1})^3 - 1\right) \tag{C.10}$$

subcubes at search depth D_r, due to the octree structure. Finally, substituting (C.1) into (C.10) and simplifying using Bachmann–Landau notation gives

$$N_r = O\left(\left(\frac{\rho_{r_0}}{\eta}\right)^3\right) = O\left(\eta^{-3}\right). \tag{C.11}$$

The ρ_{r_0} term is removed because it is a constant (equal to $\sqrt{3}\pi/2$) that is not selected by the user. □

The analysis of the worst-case search depth and time complexity for translation search proceeds in a similar manner.

Theorem C.2. *(Translation Search Depth and Complexity) Let ρ_{t_0} be the half space diagonal of the initial translation sub-cuboid $\mathcal{C}_{t_0}$. Then*

$$D_t = \max\left\{\left\lceil \log_2 \frac{\rho_{t_0}}{\zeta \sin\eta} \right\rceil, 0\right\} \tag{C.12}$$

is an upper bound on the worst-case translation tree search depth for an uncertainty angle tolerance η and $O(\rho_{t_0}^3 \zeta^{-3} \eta^{-3})$ is the time complexity of translation BB search.

Proof. Translation BB converges when $\underline{\nu}_t \geqslant \bar{\nu}_t$. This condition is met when $\psi_t'(\mathbf{p}, \mathcal{C}_t) \leqslant 0$ or equivalently $\psi_t(\mathbf{p}, \mathcal{C}_t) \leqslant \eta$ for all $\mathbf{p} \in \mathcal{P}$. This can be seen by inspecting (42) and (43) and noting that at convergence the upper and lower rotation bounds will be equal. Now for $\|\mathbf{p} - \mathbf{t}_0\| \geqslant \rho_t$, which is guaranteed for $\rho_t \leqslant \zeta$,

$$\begin{aligned}
\psi_t(\mathbf{p}, \mathcal{C}_t) &= \max_{\mathbf{t} \in \mathcal{V}_t} \angle(\mathbf{p} - \mathbf{t}, \mathbf{p} - \mathbf{t}_0) && \text{(C.13)}\\
&\leqslant \max_{\mathbf{t} \in S_t^2} \angle(\mathbf{p} - \mathbf{t}, \mathbf{p} - \mathbf{t}_0) && \text{(C.14)}\\
&= \arcsin\left(\frac{\rho_t}{\|\mathbf{p} - \mathbf{t}_0\|}\right) && \text{(C.15)}\\
&\leqslant \arcsin\left(\frac{\rho_t}{\zeta}\right) && \text{(C.16)}
\end{aligned}$$

where (C.14) follows from maximising the angle over the circumsphere S_t^2 of the cuboid instead of the vertices, (C.15) is shown in [1] with ρ_t being the half space diagonal of the translation subcuboid $\mathcal{C}_t$, and (C.16) follows from the restriction of the translation domain such that $\|\mathbf{p} - \mathbf{t}\| \geqslant \zeta$. At translation search tree depth D_t, the half space diagonal of $\mathcal{C}_{t_{D_t}}$ is given by

$$\rho_{t_{D_t}} = \frac{1}{2}\rho_{t_{D_t - 1}} = \frac{1}{2^{D_t}}\rho_{t_0}. \tag{C.17}$$

Substituting into (C.16) gives

$$\psi_t(\mathbf{p}, \mathcal{C}_{t_{D_t}}) \leqslant \arcsin\left(\frac{\rho_{t_{D_t}}}{\zeta}\right) = \arcsin\left(\frac{\rho_{t_0}}{\zeta 2^{D_t}}\right). \tag{C.18}$$

To find the worst-case translation search tree depth, the constraint $\psi_t(\mathbf{p}, \mathcal{C}_t) \leqslant \eta$ is applied, resulting in

$$\psi_t(\mathbf{p}, \mathcal{C}_{t_{D_t}}) \leqslant \arcsin\left(\frac{\rho_{t_0}}{\zeta 2^{D_t}}\right) \leqslant \eta. \tag{C.19}$$

Taking the sine and logarithm of both sides yields

$$D_t \geqslant \log_2 \frac{\rho_{t_0}}{\zeta \sin\eta}. \tag{C.20}$$

Equation (C.12) follows from the requirement that D_t be a non-negative integer. Now, translation BB will have examined at most

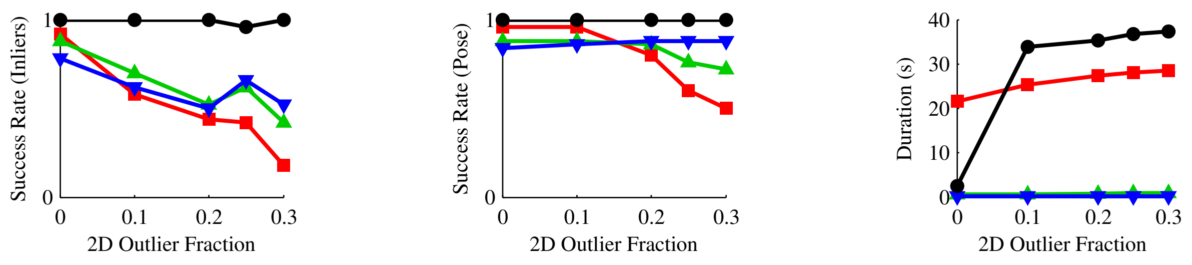
$$N_t = 8(1 + 8 + 8^2 + \cdots + 8^{D_t}) = 8\frac{8^{D_t+1} - 1}{8 - 1} = \frac{8}{7}\left((2^{D_t+1})^3 - 1\right) \tag{C.21}$$

subcuboids at search depth D_t. Finally, substituting (C.12) into (C.21) and simplifying using Bachmann–Landau notation and the Taylor expansion of $\sin\eta$ gives

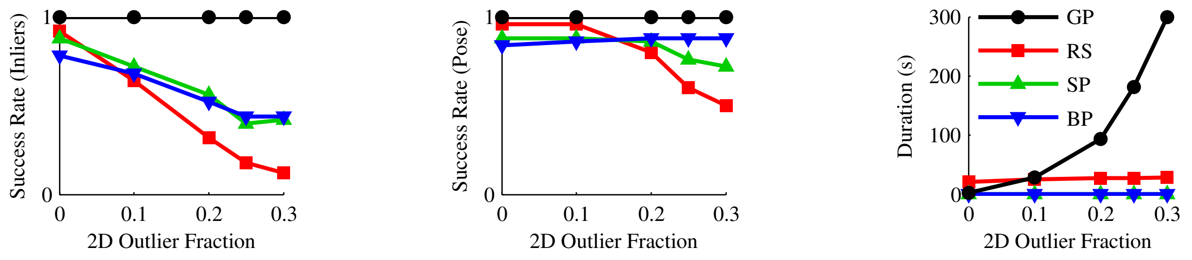
$$N_t = O\left(\rho_{t_0}^3 \zeta^{-3} (\sin\eta)^{-3}\right) = O\left(\rho_{t_0}^3 \zeta^{-3} \eta^{-3}\right). \tag{C.22}$$

□

In the nested BB search structure detailed at the beginning of Section 5, for every translation subcuboid examined, rotation BB search is run once to find the lower translation bound and again to find the upper translation bound. Thus the number of rotation subcubes examined is at worst equal to $2N_tN_r$. For each rotation subcube, both the upper and lower bounds must be calculated, each with a time complexity of $O(MN)$. Thus the total number of bound calculations is at worst equal to $4N_tN_r$. Combining the time complexity analyses (C.11) and (C.22) with the time complexity of the bound calculations leads to the following corollary.



(a) Truncated GOPAC



(b) Standard GOPAC

Figure D.1. A comparison of "truncated GOPAC" and standard GOPAC. Mean success rates and median runtimes with respect to the 2D outlier fractions for the random points dataset, for 50 Monte Carlo simulations per parameter value with the torus prior. The 3D outlier fraction was fixed at 0.

Corollary C.1. *(Time Complexity of GOPAC) Let ρ_{t_0} be the half space diagonal of the initial translation subcuboid $\mathcal{C}_{t_0}$, ζ be the translation restriction parameter, η be the uncertainty angle tolerance, M be the number of 3D points and N be the number of bearing vectors, then the time complexity of the GOPAC algorithm is given by*

$$O\left(\rho_{t_0}^3 \zeta^{-3} \eta^{-6} MN\right). \tag{C.23}$$

It is important to observe that experimental evaluation of runtime is more revealing for BB algorithms than time complexity analysis. The main reason to use BB is that it can prune large regions of the search space, reducing the size of the problem. This is not reflected in the complexity analysis.

D. Additional Experiments and Results

D.1. Synthetic Data Experiments: Early Termination

The early termination strategy, referred to in the paper as "truncated GOPAC" or $\lfloor \text{GP} \rfloor$, is investigated further in this section. Recall that since the majority of the runtime of the algorithm is spent decreasing the upper bound, an early termination strategy will often attain the global optimum, although it will not be able to guarantee optimality. We repeated the 2D outlier experiments for the torus prior with random points to show the performance of our algorithm when it is terminated after 30s. At termination, the algorithm returns the best-so-far cardinality and camera pose, as well as a flag to indicate that the result is not guaranteed to be optimal. The results are shown in Figure D.1. It is clear that the method still performs very well even when terminated early, albeit without an optimality guarantee. For some applications, it may be worth sacrificing optimality for the significant decrease in runtime. For comparison, we also plot the results from the paper without truncation.

D.2. Real Data Experiments

In this section, we expand the quantitative and qualitative results for the experiments reported in Section 6.2. In Tables D.1–D.4, Tables 1–4 have been rewritten with a tighter success rate (starred), halving the relative translation error criterion to 5%. Good performance is still achieved with this much stricter criterion, indicating that the camera poses are found to a relatively high precision. In Figure D.2, Figure 17 has been re-plotted at a larger scale. In Figures D.3, D.4 and D.5, we plot the remaining 10 images with the results obtained by GOPAC and RANSAC. It can be seen that there are two failure cases, with respect to camera pose (both remain optimal with respect to the number of inliers). In both cases, some of the extracted 2D features lie on non-building pixels, due to an error in the segmentation, which can be thought of as particularly undesirable 2D outliers. This is likely to have contributed to the algorithm finding the incorrect pose.

Table D.1. Camera pose results for serial and parallel (CPU and GPU) implementations of GOPAC and RANSAC for scene 1 of the Data61/2D3D dataset. The median translation error, rotation error and runtime and the mean inlier recall and success rates are reported. The starred success rate has double the translation precision requirements of the unstarred rate.

Implementation	Serial		Parallel: CPU		Parallel: GPU		RANSAC
Angular tolerance η	0	10^{-3}	0	10^{-3}	0	10^{-3}	–
Translation error (m)	2.30	**2.22**	2.30	2.29	**2.22**	**2.22**	28.5
Rotation error (°)	2.18	**2.08**	**2.08**	2.09	2.09	2.09	179
Recall (inliers)	1.00	1.00	1.00	1.00	1.00	1.00	0.81
Success rate (inliers)	1.00	1.00	1.00	1.00	1.00	1.00	0.00
Success rate (pose)	0.82	0.82	0.82	0.82	0.82	0.82	0.09
Success rate* (pose)	0.73	0.73	0.73	0.73	0.73	0.73	0.09
Runtime (s)	614	352	477	323	8	**6**	471

Table D.2. Camera pose results for the quad-GPU implementation of GOPAC for the Data61/2D3D dataset. The median translation error, rotation error and runtime and the mean inlier recall and success rates are reported. The starred success rate has double the translation precision requirements of the unstarred rate.

Scene	1	2	3	4	5	6	7	8	9	10
Number of 3D points	514	572	721	314	259	234	245	439	819	899
Translation error (m)	1.11	0.97	1.06	1.57	1.12	1.12	0.34	1.50	0.87	0.83
Rotation error (°)	0.70	1.45	1.51	1.36	1.15	0.84	0.59	1.40	0.83	1.45
Recall (inliers)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Success rate (inliers)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Success rate (pose)	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.91	1.00	1.00
Success rate* (pose)	1.00	1.00	0.82	1.00	1.00	1.00	1.00	0.91	1.00	1.00
Runtime (s)	15	27	11	7	11	25	7	20	25	24

Table D.3. Camera pose results for the quad-GPU implementation of GOPAC (GP) and RANSAC (RS) for area 3 of the Stanford 2D-3D-S dataset. The median translation error, rotation error and runtime and the mean inlier recall and success rates are reported. The starred success rate has double the translation precision requirements of the unstarred rate.

Room type	lounge		office		other	
Mean point-set size	534		299		365	
Method	GP	RS	GP	RS	GP	RS
Translation error (m)	**0.07**	0.68	0.18	1.85	0.13	1.87
Rotation error (°)	**1.74**	13.0	3.40	89.7	2.95	37.5
Recall (inliers)	**1.00**	0.62	**1.00**	0.63	**1.00**	0.59
Success rate (inliers)	**1.00**	0.00	**1.00**	0.00	**1.00**	0.00
Success rate (pose)	**1.00**	0.20	0.80	0.10	**1.00**	0.14
Success rate* (pose)	**1.00**	0.10	0.80	0.10	**1.00**	0.00
Runtime (s)	**12**	121	40	121	35	121

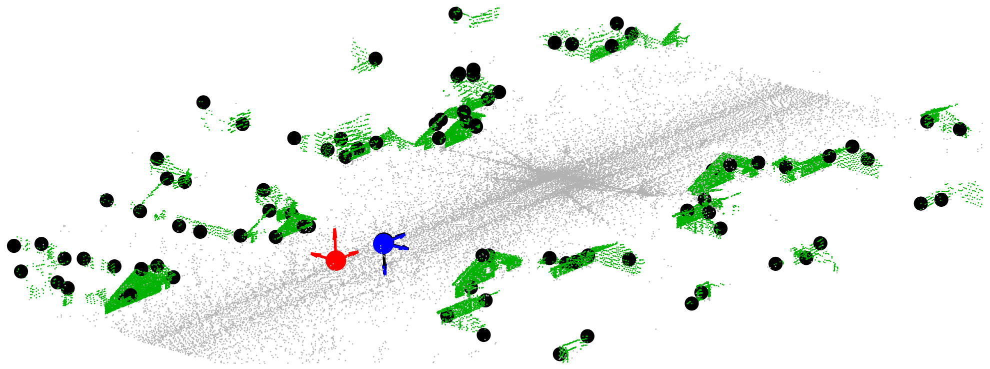
References

[1] M. Brown, D. Windridge, and J.-Y. Guillemaut. Globally optimal 2D-3D registration from points or lines without correspondences. In *Proc. 2015 Int. Conf. Comput. Vision*, pages 2111–2119, 2015. 4

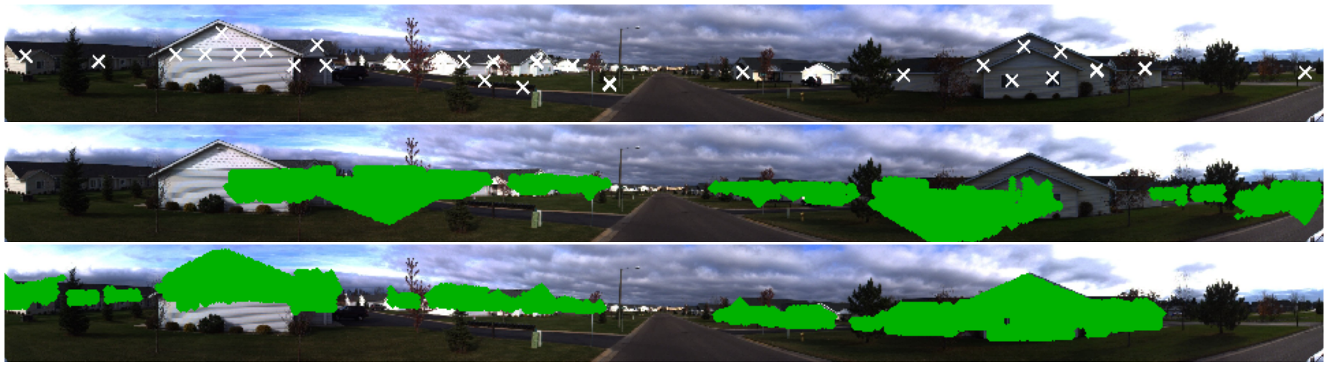
[2] G. Gallego and A. Yezzi. A compact formula for the derivative of a 3-D rotation in exponential coordinates. *J. Mathematical Imaging*

Table D.4. Camera pose results for the quad-GPU implementation of GOPAC (GP) and RANSAC (RS) for the edge-features A and B datasets. The median translation error, rotation error and runtime and the mean inlier recall and success rates are reported. The starred success rate has double the translation precision requirements of the unstarred rate.

Dataset	A		B	
Method	GP	RS	GP	RS
Translation error (m)	0.13	6.11	**0.10**	5.77
Rotation error (°)	2.90	141	**1.88**	129
Recall (inliers)	**1.00**	0.69	**1.00**	0.56
Success rate (inliers)	**1.00**	0.00	**1.00**	0.00
Success rate (pose)	0.68	0.00	**0.99**	0.00
Success rate* (pose)	0.56	0.00	**0.86**	0.00
Runtime (s)	**44**	120	52	120



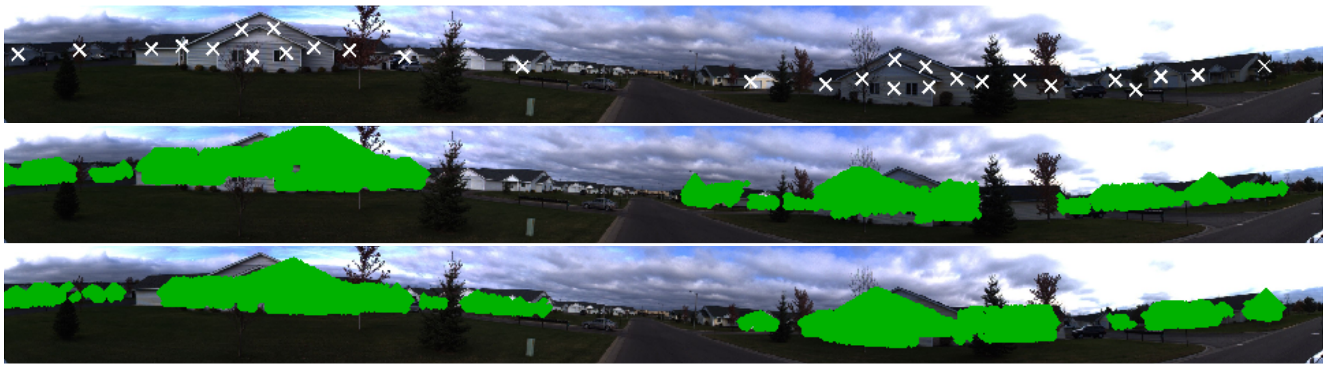
(a) 3D point-set (grey and green), 3D features (black dots) and ground-truth (black), RANSAC (red) and our (blue) camera poses. The ground-truth and our camera poses coincide, whereas the RANSAC pose has a translation offset and a 180° rotation offset. Best viewed in colour.

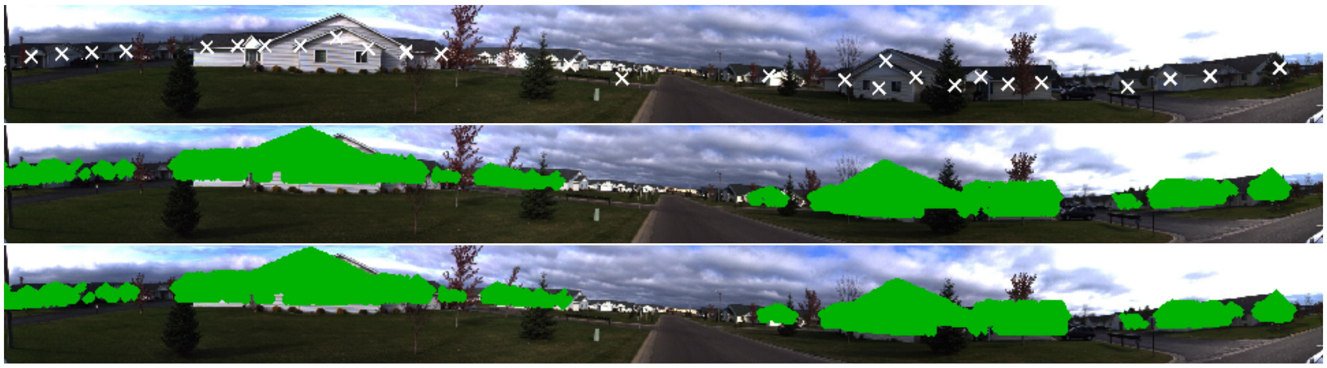


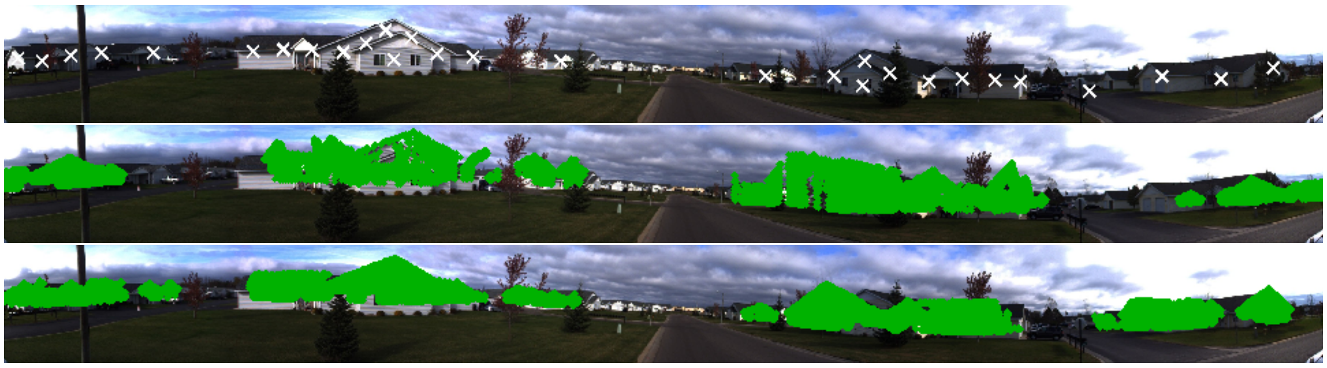
(b) Image 10. Panoramic photograph and extracted 2D features (top), building points projected onto the image using the RANSAC camera pose (middle) and building points projected using our camera pose (bottom).

Figure D.2. Qualitative camera pose results for scene 1 of the Data61/2D3D dataset, showing the pose of the camera when capturing image 10 and the projection of 3D building points onto image 10.

Vision, 51(3):378–384, 2015. 1


(a) Image 1: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).


(b) Image 2: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).


(c) Image 3: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

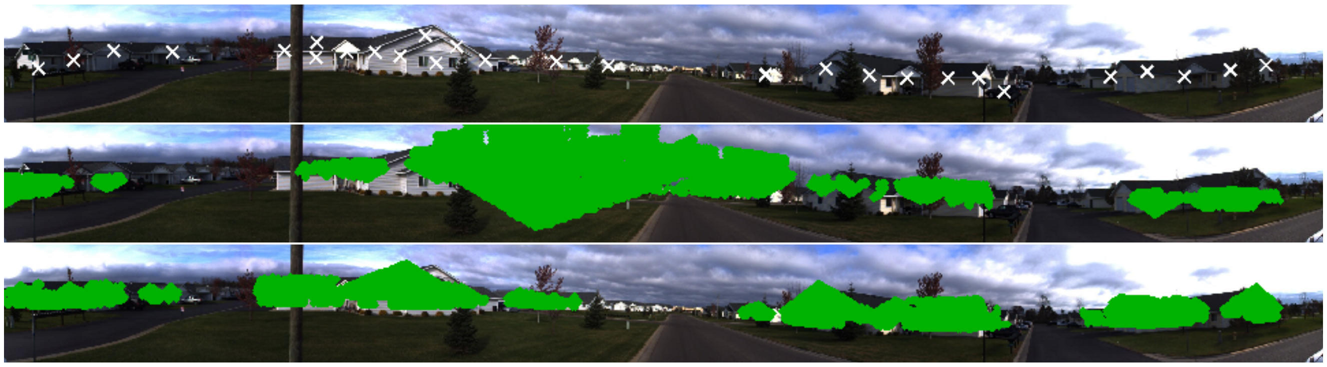
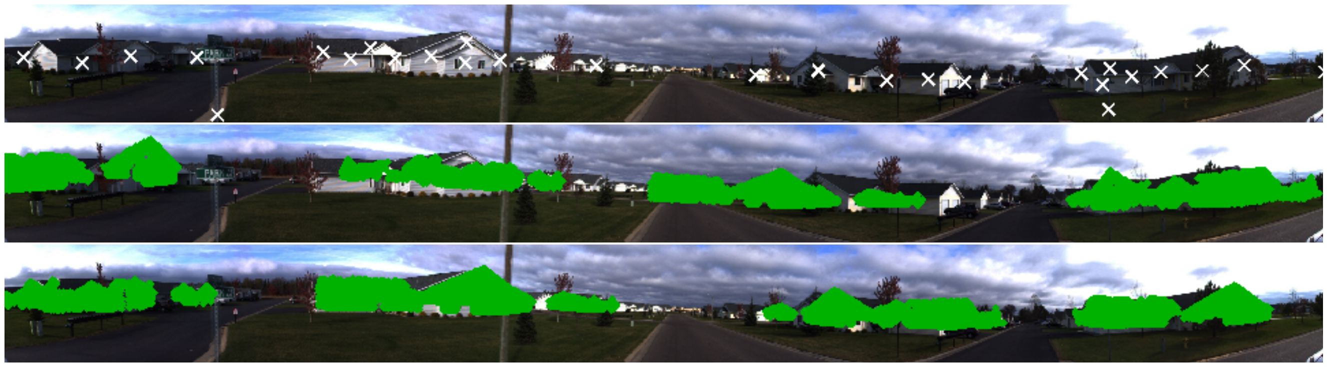
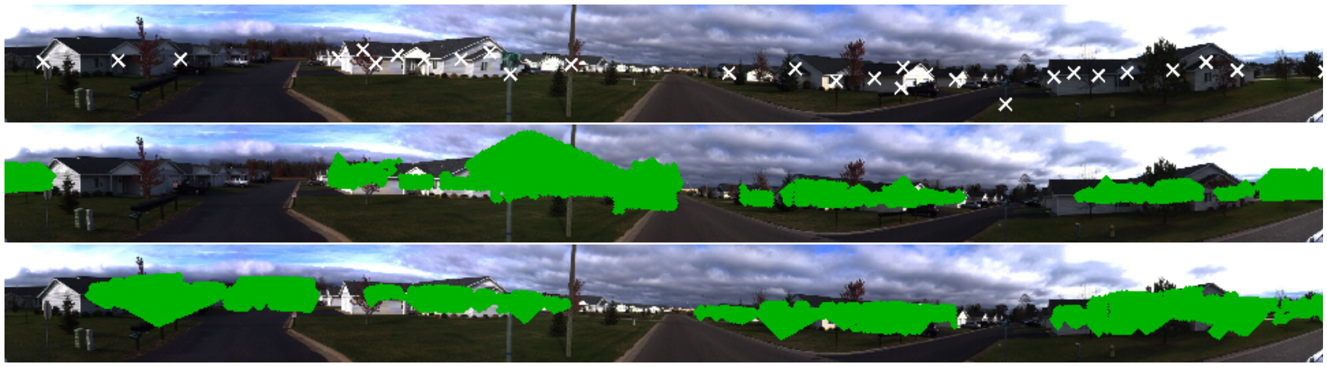

(d) Image 4: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

Figure D.3. Qualitative results for 4 test images. Our method (GOPAC) found the correct camera pose for every image.


(a) Image 5: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).


(b) Image 6: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

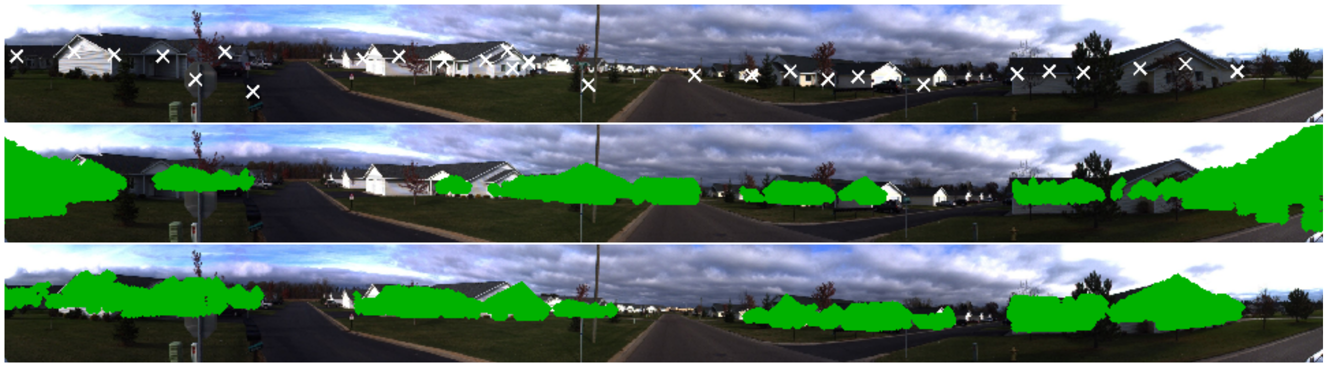
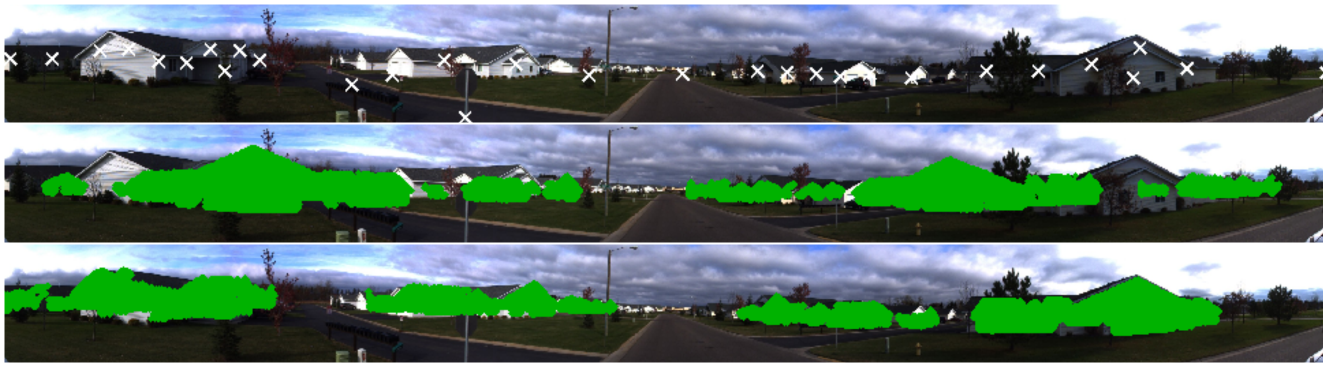
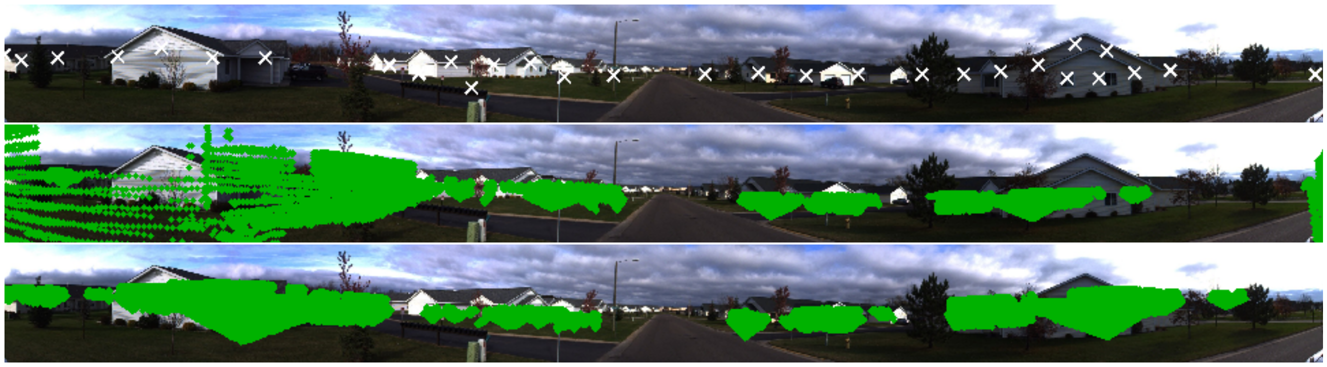

(c) Image 7: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

Figure D.4. Qualitative results for 3 test images. Image 6 was a failure case for our method, with respect to camera pose, although it still found the optimal number of inliers. It can be seen that three 2D features were extracted at non-building locations, due to an error in the segmentation. This may have contributed to the algorithm finding an incorrect pose.


(a) Image 8: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).


(b) Image 9: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

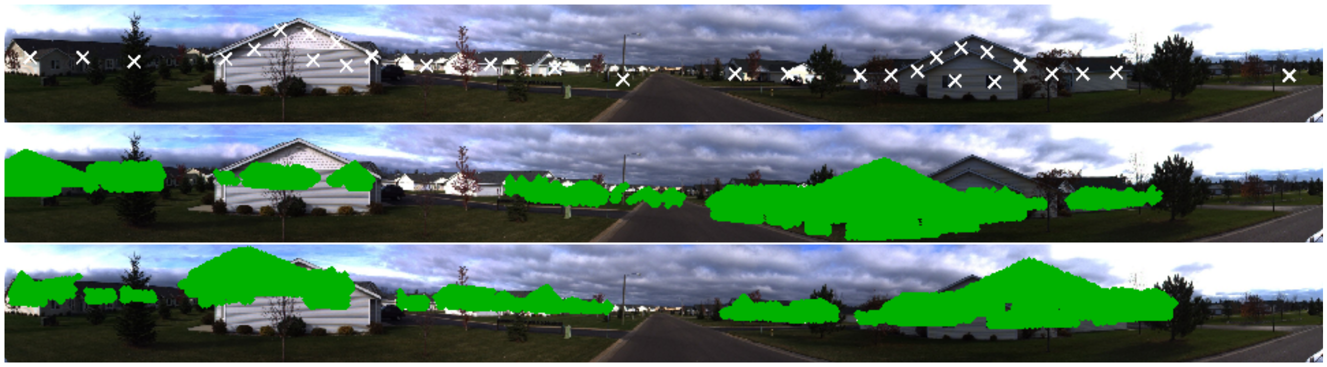

(c) Image 11: 2D features (top), building points projected using RANSAC camera pose (middle) and building points projected using our pose (bottom).

Figure D.5. Qualitative results for 3 test images. Image 9 was a failure case for our method, with respect to camera pose, although it still found the optimal number of inliers. Like the previous failure case, 2D features were extracted at non-building locations, due to an error in the segmentation. This may have contributed to the algorithm finding an incorrect pose.