Appendix A: Inference Algorithm Listing

We present here a summary of the inference algorithm for reference. It takes as input the following elements:

1. Sets of regions $P_k$, $k \in \{1, \ldots, K\}$ at $K$ different scales.
2. For each region $p \in P_k$, $k > 1$, a set $H_p$ consisting of non-overlapping child regions that partition $p$, and are from scales smaller than $k$ (i.e. $H_p \subset \bigcup_{k'<1} P_{k'}$).
3. The data cost functions $D_p(\cdot)$ and scalar outlier costs $\tau_p$ for every region $p$.
4. The value of the consistency weight $\lambda$. Additionally, its value $\lambda_0$ to be used at the beginning of the iterations, the factor $\lambda_f > 1$ by which it is to be increased at every $T_\lambda$ iterations.
5. An initial estimate $Z_0(n)$ of the scene value map.

Given these elements, the algorithm to minimize the cost function $L$ is reproduced below:

```python
# Initialization
Set $\lambda^* = \lambda_0$, $Z(n) = Z_0(n)$ for all $n$.
In parallel, for all $p \in P_1$:
    Set $Q_p = \sum_{n \in p} U(n)^T U(n)$.
end;
for $k = 2 \ldots K$
    In parallel, for all $p \in P_k$:
        Set $Q_p = \sum_{c \in H_p} Q_c$.
end;
end;

# Main Iterations
for iters = 1 \ldots MAXITERS
    # Upsweep
    In parallel, for all $p \in P_1$:
        Set $\phi_p = \sum_{n \in p} U(n)^T Z(n)$, $e_p = \sum_{n \in p} \|Z(n)\|^2$.
    end;
    for $k = 2 \ldots K$
        In parallel, for all $p \in P_k$:
            Set $\phi_p$, $e_p$ as per (10).
        end;
    end;

    # Minimize
    In parallel, for all $p \in P$:
        Set $\theta_p$, $I_p$ as per (8) and (9).
    end;

    # Downsweep
    for $k = K, K-1 \ldots 1$
        In parallel, for all $p \in P_k$:
            Set $\theta_p^+$, $I_p^+$ as per (11).
        end;
    end;
    In parallel, for all $n$:
        Set $Z(n)$ as per (12).
    end;

    # Update $\lambda^*$
    Set $\lambda^* = \text{MIN}(\lambda^* \times \lambda_f, \lambda)$ if mod(iters,$T_\lambda$) = 0.
end;
```

Appendix B: Evolution of Consensus Objective during Optimization

![Graph](image)

Figure 5. This figure shows the evolution of the consensus cost during optimization for a typical image with using different initial values \(\lambda_0\) and update schedules \(\lambda_f, T_\lambda\) (see Appendix A) for \(\lambda'\). The consensus cost shown is computed with the true value of the consistency weight \(\lambda\) (even for the iterations when minimization is done with lower values \(\lambda'\)), and the occlusion-based correction step is omitted.

As described in Sec. 4, to avoid poor local minima and promote convergence to a good solution with a low cost, we use a lower value \(\lambda'\) of the consistency weight in the early iterations of the alternating minimization method, and increase it slowly to the desired weight \(\lambda\). Figure 5 illustrates the effect of different schedules for \(\lambda'\) on convergence for a typical example. In Fig. 5 (a), we show the evolution of the objective starting with different values of \(\lambda_0 = \lambda\), and increasing it by a constant factor of \(\lambda_f = \sqrt{2}\) at every iteration, and keeping it fixed after it reaches \(\lambda' = \lambda\). We see that the direct alternating minimization case \((\lambda_0 = \lambda)\) decreases the consensus cost sharply in the first few iterations, but then stagnates at a local minima with a relatively high cost. As we lower the starting value of \(\lambda_0\), the cost has higher values and decreases more gradually in the initial iterations, but continues to decrease over a larger number of iterations and eventually converges to a better solution with a lower cost. Figure 5 (b) explores the effect of a higher rate \(\lambda_f\) of increasing \(\lambda'\). We see that like with a lower starting value for \(\lambda_0\), a slower rate \(\lambda_f\) leads to convergence to a better solution, albeit more gradually.

In addition to requiring more iterations to converge, another computational penalty of changing \(\lambda'\) across iterations is that it requires re-doing any pre-computations that depend on the consistency weight. For our stereo algorithm, minimizing the sum of the data and consistency costs involves solving a \(3 \times 3\) linear system for each region, and changing the value of \(\lambda'\) requires re-doing the LDL decompositions of the system matrices. Since this is expensive, it is desirable to avoid changing the value of \(\lambda'\) at every iteration. In Fig. 5 (c), we consider different cases with the same value of \(\lambda_0\) while jointly setting the increase factor \(\lambda_f\), and the interval \(T_\lambda\) at which it is applied, so that the total number of iterations taken for \(\lambda'\) to reach its final value \(\lambda\) remains the same \((i.e., by applying a higher rate at larger intervals)\). We see that choosing higher intervals leads to a “stair-casing” effect in the evolution of the objective, but the solution it converges to is only worse by a relatively small margin. We find this to be an acceptable trade-off between convergence to a low-cost solution and limiting computational expense, and use the parameters \(\lambda_0 = \lambda/2^{18}, \lambda_f = \sqrt{2^{18}}, T_\lambda = 6\) in our stereo implementation.