Appendix

Active Perception using Light Curtains for Autonomous Driving

Siddharth Ancha, Yaadhav Raaj, Peiyun Hu,
Srinivasa G. Narasimhan, and David Held

Carnegie Mellon University, Pittsburgh PA 15213, USA
{sancha,ryaadhav,peiyunh,srinivas,dheld}@andrew.cmu.edu

A Information gain objective

In this section, we derive the optimization objective used in Sections 4.3 and 4.4, from a perspective of maximizing information gain. Information gain is a well-defined mathematical quantity, and choosing sensing actions to maximize information gain has been used as the basis of many works on next-best view planning (see Sec. 3).

We will first describe some notation, and make two simplifying assumptions in order to derive our objective as an approximation of information gain.

A.1 Notation

- The detector predicts the probability of a detection at every anchor box location. Let there be a total of K discrete anchor box location, which are usually organized as a regular 2D-grid (see Sec. 4.2). Let $\mathbf{A}_k$ denote the k-th anchor box, where $1 \leq k \leq K$. Define $\mathbf{A} = \{\mathbf{A}_k\}_{k=1}^K$ to be the vector of all anchor boxes.
- Let $D_{\mathbf{A}_k}$ be a binary random variable denoting whether a detection exists at $\mathbf{A}_k$. $D_{\mathbf{A}_k} \in \{0, 1\}$; it is 0 if there is no detection at $\mathbf{A}_k$, and 1 if there is. Define $D_{\mathbf{A}} = \{D_{\mathbf{A}_k}\}_{k=1}^K$.
- Given a unified point cloud C, an inference algorithm (in this case, the detector) outputs a probability distribution $P(D_{\mathbf{A}} \mid C)$ over all possible detection states $D_{\mathbf{A}} \in \{0, 1\}^K$. Denote by $P(D_{\mathbf{A}_k})$ the marginal probability distribution of detection at $\mathbf{A}_k$.
- As discussed in Sec. 3, a single light curtain placement is defined by a set of control points $L = \{\mathbf{X}_t\}_{t=1}^T$. The light curtain will be placed to lie vertically on top of these control points. The 3D points sensed by this light curtain are fused back into C, to obtain an updated unified point cloud C'. We assume for now that the control points $\mathbf{X}_t$ correspond to some anchor box locations.

A.2 Assumptions

We now make the following assumptions:

1. *Detections probabilities across locations are independent.*
   That is, $P(D_{\mathbf{A}} \mid C) = \prod_{k=1}^{K} P(D_{\mathbf{A}_k} \mid C)$. This is a reasonable assumption, since the probability of detections at one location should be unaffected by detections in other locations. A consequence of this assumption is that the overall entropy $H(D_{\mathbf{A}} \mid C)$ can be written as the sum of entropies over individual anchor locations i.e. $H(D_{\mathbf{A}} \mid C) = \sum_{k=1}^{K} H(D_{\mathbf{A}_k} \mid C)$ (since the entropy of independent random variables is the sum of their individual entropies).
2. *Light curtain sensing resolves uncertainty fully but locally.*
   After placing $L = \{\mathbf{X}_t\}_{t=1}^{T}$, updating the unified point cloud to C', re-running the detector, and obtaining a new probability distribution of the updated detections $D'_{\mathbf{A}}$, the following hold.
   (a) The uncertainty of locations covered by the curtain reduces to zero: $P(D'_{\mathbf{A}_k} \mid C') \in \{0, 1\}$ for all $\mathbf{A}_k \in L$.
   (b) The uncertainty of all the other locations remains unchanged: $P(D'_{\mathbf{A}_k} \mid C') = P(D_{\mathbf{A}_k} \mid C)$ for all $\mathbf{A}_k \notin L$.

Assumptions 1 and 2 imply that the entropy of the updated distribution is given by (here K is the total number of anchor locations, and T is the number of locations that the light curtain lies on).

$$
\begin{aligned}
H(D'_{\mathbf{A}} \mid C') &= \sum_{k=1}^{K} H(D'_{\mathbf{A}_k} \mid C') \\
&= \sum_{\mathbf{A}_k \in L} \underbrace{H(D'_{\mathbf{A}_k} \mid C')}_{=\ 0 \text{ as } P(D'_{\mathbf{A}_k} \mid C') \in \{0,1\}} + \sum_{\mathbf{A}_k \notin L} \underbrace{H(D'_{\mathbf{A}_k} \mid C')}_{=\ H(D_{\mathbf{A}_k} \mid C)} \\
&= \sum_{\mathbf{A}_k} H(D_{\mathbf{A}_k} \mid C) - \sum_{\mathbf{A}_k \in L} H(D_{\mathbf{A}_k} \mid C) \\
&= \sum_{k=1}^{K} H(D_{\mathbf{A}_k} \mid C) - \sum_{\mathbf{A}_k \in L} H(D_{\mathbf{A}_k} \mid C) \\
&= H(D_{\mathbf{A}} \mid C) - \sum_{t=1}^{T} H(D_{\mathbf{X}_t} \mid C)
\end{aligned}
$$

The information gain, which is essentially a difference between the prior and updated entropies, is

$$\begin{aligned}\text{Information Gain} &= H(D_{\mathbf{A}} \mid C) - H(D'_{\mathbf{A}} \mid C') \\ &= H(D_{\mathbf{A}} \mid C) - \Big(H(D_{\mathbf{A}} \mid C) - \sum_{t=1}^{T} H(D_{\mathbf{X}_t} \mid C)\Big) \\ &= \sum_{t=1}^{T} H(D_{\mathbf{X}_t} \mid C)\end{aligned}$$

Optimization objective: This leads us to an optimization objective where maximizing information gain is equivalent to simply maximizing the sum of uncertainties (binary entropies) over the control points the curtain lies on. The maximization objective then becomes: $J(\mathbf{X}_1, \ldots, \mathbf{X}_T) = \sum_{t=1}^{T} H(\mathbf{X}_t)$, where $H(\mathbf{X})$ is the binary entropy of the detectors confidence at the location of $\mathbf{X}$.

B Hierarchical optimization objective for smoothness

Section 4.4 described an efficient algorithm for optimally placing light curtains to maximize coverage of high uncertainy regions. However, if two valid light curtain placements $\{\mathbf{X}'_t\}_{t=1}^{T}, \{\mathbf{X}''_t\}_{t=1}^{T}$ have equal sum of uncertainties, which one should we prefer? Distinct light curtain placements can have equal sums of uncertainties due to regions where the detector uncertainty is uniform. In such cases, we can choose to prefer curtains that are *smooth*, i.e. the laser angle has to change the least on average. Here, we propose a hierarchical objective function that incorporates smoothness. We then show that this also has the optimal substructure property and can be optimized in a very similar manner.

We define a hierarchical objective function that ranks two placements as follows:

$$J_H(\{\mathbf{X}'_t\}_{t=1}^{T}) \geq J_H(\{\mathbf{X}''_t\}_{t=1}^{T}) \text{ iff } \begin{cases} J(\{\mathbf{X}'_t\}_{t=1}^{T}) > J(\{\mathbf{X}''_t\}_{t=1}^{T}) \\ \text{or} \\ \begin{cases} J(\{\mathbf{X}'_t\}_{t=1}^{T}) = J(\{\mathbf{X}''_t\}_{t=1}^{T}) \\ \text{and} \\ \sum_{t=1}^{T-1} |\theta(\mathbf{X}'_{t+1}) - \theta(\mathbf{X}'_t)|^2 \\ \leq \sum_{t=1}^{T-1} |\theta(\mathbf{X}''_{t+1}) - \theta(\mathbf{X}''_t)|^2 \end{cases} \end{cases}$$

This hierarchical objective prefers light curtains that cover a higher sum of uncertainties. But if two curtains have the same sum, this objective prefers the one with a lower sum of squared laser angle deviations. We note that this hierarchical objective $J_H(\mathbf{X}_1, \ldots, \mathbf{X}_T)$ also satisfies optimal substructure. In fact, it obeys the same recursive optimality equation as Equation 6. Hence, it can be accommodated by our approach with minimal modification to our algorithm. Additionally, it can be executed with no additional overhead in $O(NTB_{\text{avg}})$ time, and leads to smoother light curtains.

C Training active detection with online light curtain data generation

Note that we use the same detector to process data from the single beam LiDAR and all subsequent light curtain placements. During training, data instances need to be sampled from the single-beam LiDAR, as well as from up to K number of light curtain placements. We choose $K = 3$ in all our experiments. Crucially, since the light curtains are placed based on the output (uncertainty maps) of the detector, the point cloud data distribution from the k-th ($1 \leq k \leq K$) light curtain placement depends on the current weights of the detector. As the weights of the detector get updated during each gradient descent step, the input training data distribution from the k-th light curtain also changes. To accomodate for non-stationary training data, we propose *training with online data-generation*. This is described in Algorithm 1.

Algorithm 1: Training with Online Light Curtain Data Generation

$W_0 \leftarrow$ initial weights of the detector
$T \leftarrow$ number of training iterations
$K \leftarrow$ number of light curtain placements
Function `InputPointCloud(`W, S, k`):`
  if $k = 0$ **then**
    $P_0 \leftarrow$ point cloud from single-beam LiDAR in scene S
    return P_0
  else
    $P_{k-1} \leftarrow$ `InputPointCloud(`W`,` S`,` $k-1$`)`
    $H \leftarrow$ uncertainty map from detector with weights W and input P_{k-1}
    $P \leftarrow$ point cloud from placing light curtain optimized for H in scene S
    $P_k \leftarrow P_{k-1} \cup P$
    return P_k
for t = *1 to* T **do**
  $S_t \leftarrow t$-th training scene
  $k_t \leftarrow$ randomly sample from $\{0, 1, \ldots, K\}$
  $P_t \leftarrow$ `InputPointCloud(`W_{t-1}, S_t, k_t`)`
  $W_t \leftarrow$ gradient descent update using previous weights W_{t-1} and input P_t
return W_T

At each training iteration t, we retrieve a scene S_t from the training dataset. To create the input point cloud, we choose to either use the single-beam LiDAR data or k light curtain placements ($1 \leq k \leq K$), each of them with equal probability. For generating the k-th light curtain data, we start with the single-beam LiDAR point cloud. Then we successively perform a forward pass through the detector network with the current weights to obtain an uncertainty map. We compute the optimal light curtain placement for this map, gather points returned from placing this curtain, and finally, fuse the points back into the input point cloud. This

cycle is repeated k times to obtain the input point cloud to train on. Generating light curtain data in such an *online* fashion ensures that the input distribution doesn't diverge from the network weights during the course of training.

Ablation experiment

Here, we perform an ablation experiment on the Virtual KITTI dataset, to evaluate the importance of training with online light curtain data generation. We first collect the entire dataset at the beginning, using the initial weights of the network. Then, we freeze this data and train the detector. The results are shown in Table 1. We see that the accuracy on light curtain data (Table 1 rows 2-4) decreases substantially to less 2%, since this data distribution diverges during training. However, the performance on single-beam LiDAR remains relatively same, since the LiDAR data distribution doesn't change. This demonstrates the importance of re-generating the training data online as the weights of the detector change.

	Virtual KITTI			
	3D mAP		BEV mAP	
	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU
Single Beam Lidar	**37.68**	**18.65**	**38.14**	**30.08**
1 Light Curtain	1.41	0.48	1.61	0.75
2 Light Curtains	0.73	0.38	1.22	0.58
3 Light Curtains	0.68	0.36	1.13	0.54

Table 1: Performance of the detector trained with single-beam LiDAR and up to three light curtains, without online training data generation. The training dataset is collected using the initial weights of the network and is fixed during the remainder of training. The light curtain performance decreases substantially.

D Datasets

D.1 Virtual KITTI dataset

Virtual KITTI is a photo-realistic synthetic video dataset designed for video understanding tasks [9]. It contains 21,160 frames (10,630 unique depth maps) generated from five different virtual worlds in urban driving settings design to closely resemble five scenes in the KITTI dataset, under different camera poses and weather conditions. It provides ground truth depth maps and 3D bounding boxes. We use four scenes (ids: 0001, 0006, 0018, 0020) as our training set, and one scene (id: 0002) as our test set.

D.2 SYNTHIA dataset

We also use the latest version of the SYNTHIA dataset [29] designed for active learning purposes. It is a large dataset containing photo-realistic scenes from urban driving scenarios, and provides ground truth depth and 3D bounding box annotations. It contains 191 training scenes (~96K frames) and 97 test scenes (~45K frames).

D.3 Evaluation metrics

We evaluate using common 3D detection metrics: mean average precision (mAP) of 3D bounding boxes (denoted as 3D mAP) and of 2D boxes in the bird's eye view (denoted as BEV mAP). We also evaluate using two different IoU overlap thresholds of 0.5 and 0.7 between detection boxes and ground-truth boxes to be considered true positives.

E Noise simulations

In order to simulate noise in the real-world sensor, we add 10% noise to the light curtain input, for varying number of light curtain placements, on the Virtual KITTI dataset. The results are shown in Table 2. The results are comparable to without noise, indicating that our method is robust to noise and is likely to transfer well to real-world data.

	Virtual KITTI							
	Without noise				**With noise**			
	3D mAP		BEV mAP		3D mAP		BEV mAP	
	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU
Single Beam Lidar	39.91	15.49	40.77	36.54	39.03	17.13	39.93	30.26
1 Light Curtain	58.01	35.29	58.51	47.05	57.04	25.99	57.65	45.31
2 Light Curtains	60.86	37.91	61.10	49.84	59.43	30.91	59.89	46.11
3 Light Curtains	**68.52**	**38.47**	**68.82**	**50.53**	**60.02**	**31.09**	**66.78**	**46.39**

Table 2: Performance of detectors trained with single-beam LiDAR and up to three light curtains, with 10% additional noise in the light curtain input. Performance is not significantly lower than without noise.

F Efficiency analysis

In this section, we report the time taken by our method, for varying number of light curtain placements, and for different light curtain placement algorithms, in Table 3. The time (in seconds) includes the time taken for all preceding steps. For example, the time for 2 light curtain placements includes the time required for generating the single-beam LiDAR data, computing the optimal first and second light curtain placements, and all intermediate forwarded passes through the detection network while generating uncertainty maps. The time is averaged over 100 independent trials over different scenes, and we report the 95% confidence intervals.

	Single-beam LiDAR	One light curtain	Two light curtain	Three light curtain
Random	0.096 ± 0.001	0.763 ± 0.008	1.441 ± 0.014	2.133 ± 0.014
Fixed depth - 15m	0.090 ± 0.002	0.765 ± 0.008	1.412 ± 0.012	2.028 ± 0.018
Fixed depth - 30m	0.095 ± 0.002	0.789 ± 0.005	1.474 ± 0.008	2.180 ± 0.013
Fixed depth - 45m	0.094 ± 0.001	0.778 ± 0.003	1.475 ± 0.013	2.174 ± 0.012
Greedy Optimization (Randomly break ties)	0.092 ± 0.000	0.825 ± 0.014	1.547 ± 0.023	2.250 ± 0.030
Greedy Optimization (Min laser angle change)	0.086 ± 0.001	0.824 ± 0.010	1.543 ± 0.020	2.242 ± 0.028
Frontoparallel + Uncertainty	0.091 ± 0.001	0.441 ± 0.003	0.807 ± 0.006	1.165 ± 0.008
Dynamic Programming	0.097 ± 0.008	0.944 ± 0.010	1.767 ± 0.015	2.600 ± 0.020

Table 3: Time efficiency (in seconds) for varying number of light curtains and different light curtain placement algorithms. Time is averaged over 100 independent trials over different scenes, and we report the 95% confidence intervals.

Note that as we place more light curtains, more time is consumed for the network's forward pass and in calculating where to place the light curtain. This presents a speed-accuracy tradeoff; more light curtains will improve detection accuracy at the expense of taking more time. On the other hand, our method can run faster using fewer light curtains but with a decreased accuracy. This tradeoff is visualized in Figure 1.

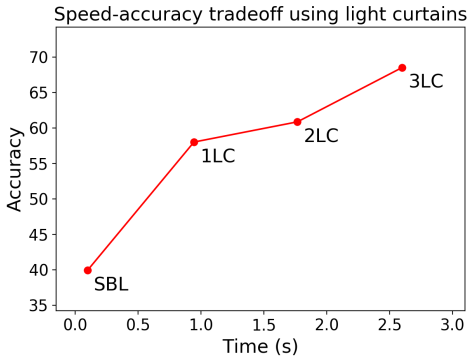

Fig. 1: Speed-accuracy tradeoff using light curtains optimized by dynamic programming, on the Virtual KITTI dataset. More light curtains correpsond to increased accuracy but reduced speed.