Synthesizing Open Worlds with Constraints using Locally Annealed Reversible Jump MCMC – Supplemental Materials

Yi-Ting Yeh
Stanford University

Lingfeng Yang
Stanford University

Matthew Watson
Stanford University

Noah D. Goodman
Stanford University

Pat Hanrahan
Stanford University

In this document, we give the formulations of the distributions used in the main text.

1 Formulating Soft Constraints

Most of the constraints used in the examples employ modulated Gaussian and sigmoid functions to express soft constraints over real-valued scoring functions (Figure 1). Here, we describe them in detail.

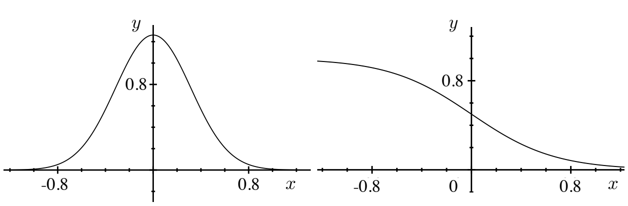
Let $\mathcal{N}(x, \mu, \sigma^2)$ be a Gaussian density with mean μ and variance σ^2 evaluated at x, and $\mathrm{Sig}(x; h) = \frac{1}{1+e^{-hx}}$ be a sigmoid function with h controlling the steepness. We formulate soft versions of logical predicates over real numbers as follows. All such values are computed in log space.

$$\begin{aligned}
\mathrm{Eq}(x, y, \sigma^2) &= \frac{\mathcal{N}(0, \|x-y\|, \sigma^2)}{\mathcal{N}(0, 0, \sigma^2)} \\
\mathrm{Greater}(x, y, h) &= \mathrm{Sig}(x-y; h) \\
\mathrm{Less}(x, y, h) &= \mathrm{Sig}(y-x; h) \\
\mathrm{Range}(x, y, z, h) &= \mathrm{Greater}(x, z, h)\mathrm{Less}(y, z, h).
\end{aligned}$$

The composition of several such functions yields a constraint whose values range from 0 (maximally unsatisfied) to 1 (maximally satisfied).

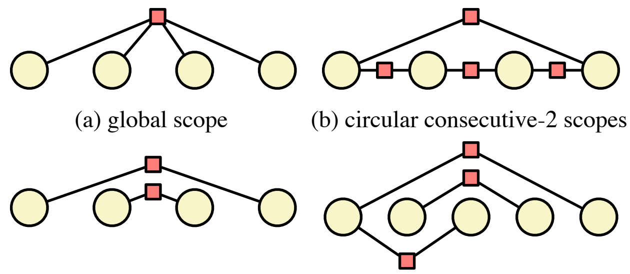
2 Simple String Example

Here, we give more details on the synthetic distributions over strings used in the paper to compare statistical efficiency of LARJ-MCMC versus other algorithms. We used two such synthetic distributions, both over strings $\mathbf{S} = S_1 S_2 \ldots S_N$ of different lengths consisting of random characters. It is feasible to calculate the normalization constant of these distributions analytically. Following are descriptions of the domains and factors of each distribution. Let N be the current number of letters in the string.



(a) $y = \mathcal{N}(x; 0, 0.1)$ (b) $y = \mathrm{Sig}(x; 3.0)$

Figure 1: *Gaussian and sigmoid functions used in designing soft constraints.*



(a) global scope (b) circular consecutive-2 scopes

(c) opposing scopes (even) (d) opposing scopes (odd)

Figure 2: *Visualization of the scopes being used in the simple string example.*

2.1 A distribution with global constraints

Domain. $\{$`'a'`,`'b'`$\}$. N can be $5 \sim 10$.

Scope and factor. There is one global factor whose scope is the entire string $\mathbf{S}$ (see Figure 2(a)). This factor constrains the string to be all `a`'s or all `b`'s depending on the length:

$$f(\mathbf{S}) = \begin{cases} Eq(\|\mathbf{S}\|, \|\{S_i | S_i = \texttt{'a'}\}\|, 0.2) & \text{if } \|\mathbf{S}\| \text{ is odd} \\ Eq(\|\mathbf{S}\|, \|\{S_i | S_i = \texttt{'b'}\}\|, 0.2) & \text{if } \|\mathbf{S}\| \text{ is even} \end{cases}$$

2.2 A distribution with local constraints

Domain. $\{$`'a'`,`'b'`,`'c'`$\}$. N can be $6 \sim 9$.

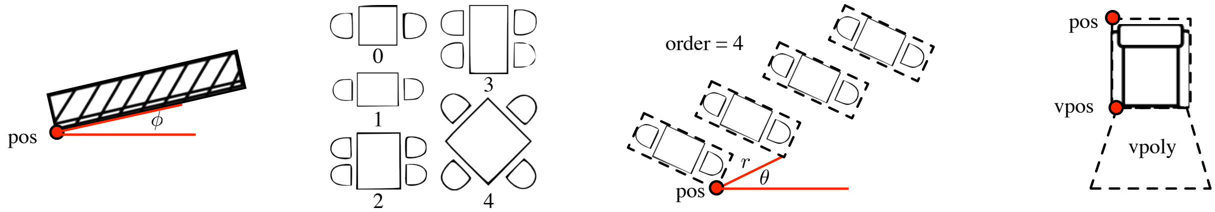
Scopes and factors.

- One factor is applied over each pair of circular consecutive characters $\{S_i, S_{i+1 \mod N}\}_{i=0}^{N-1}$. This scoping is illustrated in Figure 2(b). This factor constrains each such pair to be different, using the following factor function:

$$f(S_i, S_j) = \begin{cases} 0 & \text{if } S_i \neq S_j \\ \log(0.2) & \text{if } S_i = S_j \end{cases}$$

- Another factor is applied over pairs of opposing characters $\{S_i, S_{N-i-1}\}_{i=0}^{\lfloor N/2 \rfloor -1}$ (with an additional pair $\{S_0, S_{\lfloor N/2 \rfloor}\}$ if N is odd). This scoping is illustrated in Figure 2(c)-(d). The factor constrains all such pairs to be the same, using the following factor function:

$$f(S_i, S_j) = \begin{cases} 0 & \text{if } S_i = S_j \\ \log(0.05) & \text{if } S_i \neq S_j \end{cases}$$



(a) shelf (b) types of tables (c) table group (d) sofa

Figure 3: *Visualization of variables used in cafe layout example.*

3 Cafe Layout

3.1 Random variables

Each piece of furniture or a group of tables O contains the following attributes:

- O.pos $\in \mathbb{R}^2$ (Figure 3(a, c, d)).
- O.orientation $= \phi \in \{0, \frac{1}{12}\pi, \ldots, 2\pi\}$ for table groups, $\{0, \frac{1}{8}\pi, \ldots, 2\pi\}$ for sofas and shelves. (see Figure 3(a)).
- O.type $\in \{0, \ldots, 4\}$ for table groups, $\{0, 1\}$ for sofas. (see Figure 3(b)).

Each group of tables also includes the following attributes (Figure 3(c)):

- O.offset $= (r, \theta) \in \mathbb{R}^+ \times \{0, \frac{1}{4}\pi, \ldots, 2\pi\}$.
- O.order $\in \{2, \ldots, 10\}$.

Sofas contain these additional attributes (Figure 3(d)):

- O.vpoly to denote view area.
- O.vpos to denote position of view area.

Figure 3 shows how these parameters affect the layout visually.

3.2 Constraint specification

Figure 4 shows the factor functions used in the cafe layout example. The following is pseudocode for certain functions used in the constraints.

```
// bbox: gets the axis-aligned bounding box of an object.
// rotate_bbox: rotates the points of a bbox about the
     origin by the given angle.
Spacing(O) {
    b = bbox(rotate_bbox(bbox(O), O.orientation));
    r, theta = O.offset;
    spacing = max(abs(b.width - r * cos(theta)), abs(b.
        height - r * sin(theta)))
    return spacing;
}
```

4 Golf Course Layout

4.1 Random variables

Each course is parameterized by the following entities (see Figure 5):

- The path of the course is a list of P_i (two points per hole; 18 points in a 9 hole course)
- Each hole contains a start, middle, and end control point. The start and end of each hole correspond to a pair of path points P_i.
- Greens and fairways are defined by blob control point objects, each of which has a radius and position attribute.
- Flags F_i contain a single position attribute.
- Traps T_i are a collection of blob control points, all sharing a common radius.

4.2 Constraint specification

Figure 6 contains a list of factor functions used in the golf course layout example. The following is pseudocode for the functions used. `Straight(p1, p2, p3)` computes the magnitude of the mean curvature for a discrete curve segment [Sullivan 2006].

```
OutsideBlob(pos, blob, size_param){
  return Less(0.5, ImplicitShapeFn(pos, blob.
       control_points, size_param), 100);
}

InsideBlob(pos, blob, size_param){
  return Greater(0.5, ImplicitShapeFn(pos, blob.
       control_points, size_param), 100);
}

ImplicitShapeFn(pos, control_points, size_param){
  val = 1.0;
  for(cp in control_points){
    dist = d(pos, cp.pos);
    val -= (4/9)*(dist/( cp.rad + size_param))^6 - (17/9)
        *(dist/(cp.rad + size_param))^4 + (22/9)*(dist/(
        cp.rad + size_param))^2;
  }
  return val;
}

AvoidIntersection(s1, s2, line_width){
  a1 = s1.first; a2 = s1.second;
  b1 = s2.first; b2 = s2.second;
  if(Intersecting(a1, a2, b1, b2)){
    cross = IntersectionPoint(a1, a2, b1, b2);
    amid = Midpoint(a1, a2);
    bmid = Midpoint(b1, b2);
    dist_to_midpts = d(cross, amid) + d(cross, bmid);
    max_dist_to_midpts = d(a1, amid) + d(b1, bmid);
    intersect_factor = 2 - dist_to_midpts /
        max_dist_to_midpts; // more intersection if
        closer to midpoint
    Greater(intersect_factor * 2 * line_width, 0, 1.0);
  }
  else{ // mutual minimum distance from pt to line
    dist = min(DistPtLine(a1, b1, b2), DistPtLine(a2, b1,
        b2),
             DistPtLine(b1, a1, a2), DistPtLine(b2, a1, a2)
                 );
    Greater(2 * line_width, dist, 1.0);
  }
}
```

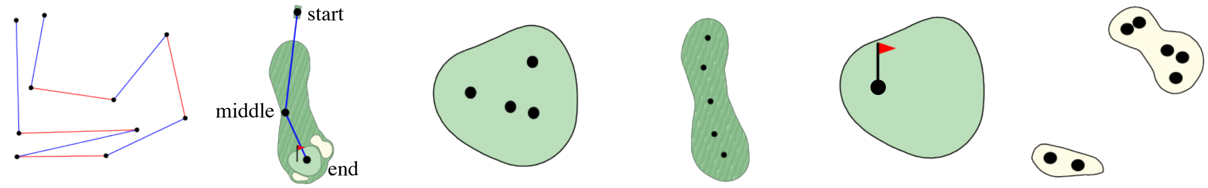
```
Straight(p1, p2, p3) {
   return discrete_curvature(p1, p2, p3);
}
```

References

SULLIVAN, J. M. 2006. Curvature measures for discrete surfaces. In *ACM SIGGRAPH 2006 Courses*, ACM, New York, NY, USA, SIGGRAPH '06, 10–13.

Description	Scope	Factor Function	Weight
non-overlap	each object pair O_i, O_j	$Eq\left(0, \frac{A(P(O_i) \cap P(O_j))}{\min\{A(P(O_i)), A(P(O_j))\}}, 0.1\right)$	40
inside room R	each object O_i	$Eq\left(0, \frac{A(P(O_i) - P(R))}{A(P(O_i))}, 0.1\right)$	40
align to nearest segment W	each table group, shelf, and plant O_i	$Eq\left(0, \sin(\Theta(O_i, W)), 0.1\right)$	40
spacing within group	each table group O_i	$Eq\left(80, \text{Spacing}(O_i), 400\right)$	30
distance to nearest segment W	each shelf O_i	$Eq\left(0, d_s(O_i.\text{pos}, W), 10^4\right)$	40
occupied area in zone Z	all sofas $\mathbf{O}$	$Eq\left(0.6, \frac{A((\cup P(O_i)) \cap P(Z))}{A(P(Z))}, 0.2\right)$	30
occupied area in Room R	all table groups $\mathbf{O}$	$Eq\left(0.6, \frac{A((\cup P(O_i)) \cap P(R))}{A(P(R))}, 0.1\right)$	30
total number of different types	all table groups $\mathbf{O}$	$Eq(2, \|\{O_i.type \| O_i \in \mathbf{O}\}\|, 1)$	30
balance w.r.t. point c	all table groups $\mathbf{O}$	$Eq(0, d(\frac{\sum A(P(O_i))O_i.\text{pos}}{\sum A(P(O_i))}, c), 10^5)$	30
encourage conversation	each sofa pair O_i, O_j	If $d(O_i, O_j) < 500$, then $Eq\left(1, \frac{A(O_i.\text{vpoly} \cap O_j.\text{vpoly})}{A(O_i.\text{vpoly})}, 0.1\right)$	1
closeness 1	each sofa pair O_{2i}, O_{2i+1}	$Range\left(100, 500, d(O_{2i}.\text{pos}, O_{2i+1}.\text{pos})\right)$	1
closeness 2	each sofa pair O_{2i}, O_{2i+1}	$Eq\left(0, d(O_{2i}.\text{vpos}, O_{2i+1}.\text{vpos}), 200\right)$	1
inside sofa zone Z	each sofa O_i	$\max\{Less\left(0, d_{poly}(O_i.\text{pos}, Z)\right), Less\left(0, d_{poly}(O_i.\text{endpos}, Z)\right)\}$	1
close to sofas $\mathbf{S}$	all lamps $\mathbf{O}$	$Less\left(75, \min_{S_i \in \mathbf{S}} d_{poly}(O_i.\text{pos}, P(S_i)), 5\right)$	1
cover all tables, sofas, shelves $\mathbf{S}$	all plants $\mathbf{O}$	$\sum_{S_i \in \mathbf{S}} Range\left(-1.0, \max_{O_i \in \mathbf{O}} d_{poly}(O_i.\text{pos}, P(S_i)), 4\right)$	1
cove all sofas $\mathbf{S}$	all lamps $\mathbf{O}$	$\sum_{S_i \in \mathbf{S}} Range\left(-1.0, \max_{O_i \in \mathbf{O}} d_{poly}(O_i.\text{pos}, P(S_i)), 4\right)$	1

Figure 4: *Factor specification for cafe layout examples. $P(O)$ returns the bounding polygon associated with object O, $\Theta(X, Y)$ returns the difference in angle between line segment Y (relative to a global coordinate frame) and the orientation attribute of object X. $A(x)$ returns the area of polygon x, and $\cup, \cap$ denote union/intersection operations on polygons. $d(x, y)$ returns the distance from point x to point y. $d_s(x, s)$ returns the distance from point x to line segment s. $d_{poly}(x, p)$ returns the squared distance from a point x to a polygon p, multiplied by -1 if the point is inside. $Spacing(O)$ denotes the internal spacing in a table group O, given in the pseudocode. The result of each factor function is multiplied in log space by its corresponding weight.*



(a) path (b) hole (c) green (d) fairway (e) flag (f) sand traps

Figure 5: *Visualization of variables used in the golf course layout example.*

Description	Scope	Factor Function	Weight
inside course C	each control point P_i of path	$Greater\,(30, DistPoly(P_i, C))$	1
outside lakes L	each flag and each control point P_i of path, green, and sandtrap	$OutsideBlob\,(P_i, L, 30)$	1
distance to target location t_i	each path end point P_i	$Less\,(20, d(t_i, P_i), 5)$	1
non-overlap	path segment pairs S_i and S_j	$AvoidIntersection\,(S_i, S_j, 40)$	5
par distribution for par 3, 4, 5	for all path segments $\mathbf{S}$. t_n is the target number of paths with par n	$\sum_{n=3}^{5} Eq\,(t_n, \|\{s_i \in \mathbf{S} \| par(s_i) = n\}\|, v_n)$	1
total par count	all path segments $\mathbf{S}$	$Eq\left(36, \sum_{s_i \in \mathbf{S}} par(s_i), 5\right)$	1
length range $d_{i,min}, d_{i,max}$	each path segment S_i	$Range\,(d_{min}, d_{max}, L(S_i), 1)$	0.25
no sharp bends	each consecutive 3 path control points P_i, P_j, P_k	$Less(1.2, Straight(P_i, P_j, P_k), 100)$	1
hole straightness	each hole H_i	$Eq(0, Straight(H_i.\text{start}, H_i.\text{mid}, H_i.\text{end}), 0.2)$	1
target size range $r_{i_{min}}, r_{i_{max}}$	each control point radius R_i of green and fairway	$Range(r_{i,min}, r_{i,max}, R_i, 1)$	1
distance to hole H_i	each control point P_i of green	$Less\,(15, d(H_i.\text{end}, P_i), 1)$	1
inside green G_i	each flag F_i	$InsideBlob\,(F_i, G_i, 30)$	1
fairway target location t_i	each fairway end point P_i	$Eq\,(0, d(t_i, P_i), 15)$	1
distance to hole path H_j	each fairway control point P_i	$Eq\,(0, d_{path}(P_i, H_j), 50)$	1
distance to hole path H_j	each sand trap control point P_i	$Range\,(17, 37, d_{path}(P_i, H_j), 15)$	1
outside green G_i	each control point of sand trap P_i	$OusideBlob\,(P_i, G_i, 60)$	1
sand trap location t_i	each sand trap control point P_i	$Eq\,(50, d(t_i, P_i), 2500)$	1
distance between traps	each sand trap pairs T_i, T_j	$Greater\,(50, d(C(T_i), C(T_j)), 1)$	1

Figure 6: *Factor functions used in the golf course layout example.* $d(x, y)$ *denotes distance between two points.* $d_{path}(x, y)$ *denotes the distance from a point to a path.* $C(x)$ *denotes the centroid of the control points of object* x.