Contents

```
function [pos, map, slam_state] = step_slam( data, slam_state, map, params )
```

```
%
```

(1) a priori estimate for particles using odometry

```
a_priori_particles = a_priori( slam_state, data, params );
slam_state.particles = a_priori_particles;
```

```
Error using step_slam (line 4)
Not enough input arguments.
```

(2) a posteriori estimate for particles and map using scan matching

```
[a_posteriori_weights, a_posteriori_map] = a_posteriori( slam_state, map, data, params );
```

(3) re-sample particles

```
%{
if Neff < alpha * num_particles
    particles = resample_particles( particles, a_posteriori_weights, params );
end
%}
```

Package output and return.

```
map = a_posteriori_map;
slam_state.weights = a_posteriori_weights;
slam_state.time = data.ts;
pos = slam_state.particles * slam_state.weights';   % weighted average
```