The particle filter contains N particles where each particle is described as

$$\{w^{[n]}, \vec{x}^{[n]}, \mathcal{M}^{[n]}\}_{n=1}^{N} \tag{1}$$

where $w^{[n]}$ is the weight associated to each particles

Per particle state vector is

$$\vec{x}^{[n]} = \begin{bmatrix} \vec{pos} \\ \vec{quat} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} \tag{2}$$

Per particle map probability hypothesis density (PHD) is

$$\mathcal{M}^{[n]} = \{\eta^{[n],[j]}, \mu^{[n],[j]}, P^{,[n],[j]}\}_{j=1}^{J} \tag{3}$$

where J is the number of gaussian mixture (GM) component in the PHD, $\eta^{[n],[j]}, \mu^{[n],[j]}, P^{[n],[j]}$is the intensity, mean (position) and covariance of the jth GM (or landmark) associated with the nth particle

The odometry input of the 6-DoF system is

$$\vec{u} = \begin{bmatrix} \delta x \\ \delta y \\ \delta z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{4}$$

which are the relative translation and rotation from consecutive frames. This can be obtained from visual odometry libraries such as LibVISO2 which uses RANSAC. This odometry provide an initial guess of how the camera moved but the PHD-SLAM improves the odometry estimate further.

The 6-DoF motion model $f(\vec{x}_k, \vec{u}_k)$ is defined as

$$\vec{quat}_{k|k-1} = \vec{quat}_{k-1} + 0.5 * T * quat(0, \omega_x, \omega_y, \omega_z) * \vec{quat}_{k-1} \tag{5}$$

$$\vec{quat}_{0.5} = SLERP(\vec{quat}_{k-1}, \vec{quat}_{k|k-1}, 0.5) \tag{6}$$

$$\vec{pos}_{k|k-1} = \vec{pos}_{k-1} + \vec{quat}_{0.5} * \left[\delta x, \delta y, \delta z,\right]^{T} * \vec{quat}_{0.5}^{-1} \tag{7}$$

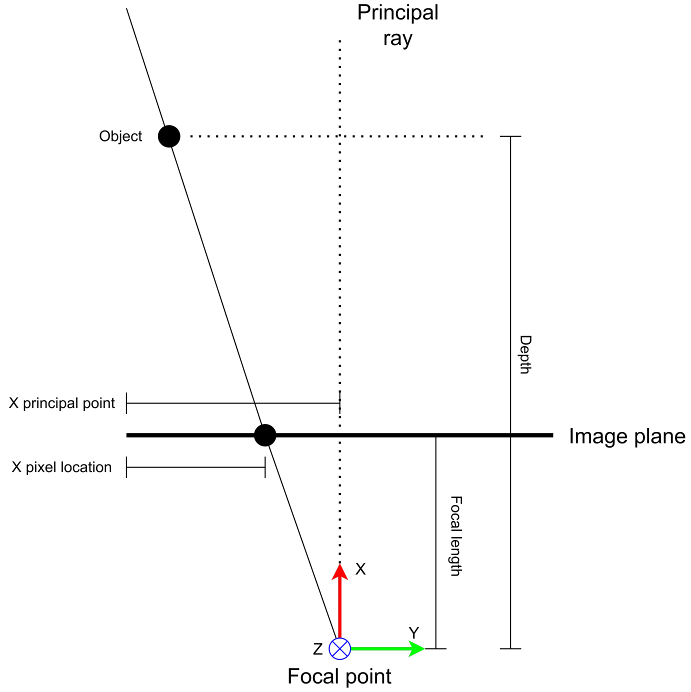

Figure 1: Measurement model

The 3D measurement model is described as below using the pinhole depth camera model. This describes the measurement generate on the rectified RGB image and depth map.

$$\vec{z}_k = \begin{bmatrix} pix_x \\ pix_y \\ d \end{bmatrix} = h(\vec{l}, \vec{x}) = \begin{bmatrix} c_x + f * \frac{Y^{cam}}{X^{cam}} \\ c_y + f * \frac{Z^{cam}}{X^{cam}} \\ X^{cam} \end{bmatrix} \tag{8}$$

where pix_x, pix_y are the location of the landmark in the image plane in x and y axis, respectively. d is the depth of the corresponding landmark in the depth map.c_x, c_y are the principal point location in the image plane in the x and y axis, respectively. f is the focal length. $\begin{bmatrix} X^{cam} & Y^{cam} & Z^{cam} \end{bmatrix}^T$ is the landmark location in the camera coordinate frame which is calculated as

$$\begin{bmatrix} X^{cam} \\ Y^{cam} \\ Z^{cam} \end{bmatrix} = R(\vec{quat}) * \begin{bmatrix} \vec{l} - \vec{pos} \end{bmatrix} = R(\vec{quat}) * \begin{bmatrix} l_x - x \\ l_y - y \\ l_z - z \end{bmatrix} \tag{9}$$

$$R(\vec{quat}) = 2* \begin{bmatrix} 0.5 - q_y^2 - q_z^2 & q_x q_y + q_w q_z & q_x q_z - q_w q_y \\ q_x q_y - q_w q_z & 0.5 - q_x^2 - q_z^2 & q_y q_z + q_w q_x \\ q_x q_z + q_w q_y & q_y q_z - q_w q_x & 0.5 - q_x^2 - q_y^2 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{10}$$

During the inner PHD update, only the Jacobian of h with respect to $\vec{l}$ is needed which is evaluated as

$$H = \frac{\delta h(\vec{l}, \vec{x})}{\delta \vec{l}} = \begin{bmatrix} f * \frac{R_{21}*X^{cam} - R_{11}*Y^{cam}}{X^{cam}*X^{cam}} & f * \frac{R_{22}*X^{cam} - R_{12}*Y^{cam}}{X^{cam}*X^{cam}} & f * \frac{R_{23}*X^{cam} - R_{13}*Y^{cam}}{X^{cam}*X^{cam}} \\ f * \frac{R_{31}*X^{cam} - R_{11}*Z^{cam}}{X^{cam}*X^{cam}} & f * \frac{R_{32}*X^{cam} - R_{12}*Z^{cam}}{X^{cam}*X^{cam}} & f * \frac{R_{33}*X^{cam} - R_{13}*Z^{cam}}{X^{cam}*X^{cam}} \\ R_{11} & R_{12} & R_{13} \end{bmatrix} \tag{11}$$

The general pseudocode for the particle filter SLAM is

Algorithm 1 PHD-SLAM1

```
Initialize
$\{w_1^{[n]}, \vec{x}_1^{[n]}, \mathcal{M}_1^{[n]}\}_{n=1}^N$ where the initial map can be constructed from the inverse model of sensor measurements
for $k = 2...t_{end}$ do
  for n = 1 ... N particles do
    Particle filter time update: propagate the sensor state $\vec{x}^{[n]}$ using $\vec{x}_k^{[n]} = f(\vec{x}_{k-1}^{[n]}, \tilde{u}_k)$ where $\tilde{u}_k \sim \mathcal{N}(u_k, Q)$
    Find previous GM in current FOV: Create a set containing only the mapped PHD (including intensity, mean, and covariance) that is in the camera current FOV
    PHD time update
    PHD measurement update
    PHD cleanup
    Add GM outside of FOV: Concatenate GM that are outside of FOV with the updated GM after PHD filter has been applied
    Single Cluster Likelihood
  end for
  Normalize weights
  State and map estimation: Use the maximum likelihood estimation
  Low variance resample
  Add birth GM
end for
```

PHD time update

After finding the GM components that would be in the FOV after the particle filter time update step, the PHD filter is applied to those GM component. Here, only the static GM component between time-step so their positions are unchanged. The process noise covariance update is performed by adding a small process covariance Q.

Algorithm 2 PHD time update

```
for $j = 1...J_{k-1}^{[n]}$ do
    $\eta_{k|k-1}^{[n],[j]} = \eta_{k-1}^{[n],[j]}$
    $\mu_{k|k-1}^{[n],[j]} = \mu_{k-1}^{[n],[j]}$
    $P_{k|k-1}^{[n],[j]} = P_{k-1}^{[n],[j]} + Q$
end for
```

PHD measurement update

Perform PHD measurement update exactly the same as a normal PHD update procedure.

PHD cleanup

The PHD cleanup step also follows the typical GM-PHD clean up procedures. First the PHD is pruned where we remove the GM components that has intensity below a cut-off threshold (T_η). Then nearby GM components are merged based on a merge threshold (T_{merge}). Then, the PHD table is truncate to only maintain a fixed limit number (J_{cap}) of highest intensity GM components.

Single Cluster Likelihood

The per particle likelihood estimate is estimated using the single cluster likelihood function

$$w_k^{[n]} = \exp(m_{k-1}^{[n]}) \times \prod_{z \in \mathbf{Z}_k} (\kappa(z) + P_d \sum_{j=1}^{J^{[n]}} \eta^{[n],[j]} p(z|\mathcal{M}^{[n],[j]}, x_k^{[n]})) \times w_{k-1}^{[n]} \tag{12}$$

Low variance resample

After the particle likelihood is calculated, the weight is normalized. We only wish to resample only when needed ie. when the number of effective particle is below a pre-defined threshold (T_{eff}). We used the low-variance method to resample.

Algorithm 3 PHD measurement update

```
Update mis-detected components
for $j = 1...J_{k-1}^{[n]}$ do
    $\eta_k^{[n],[j]} = (1 - P_d) * \eta_{k|k-1}^{[n],[j]}$
    $\mu_k^{[n],[j]} = \mu_{k|k-1}^{[n],[j]}$
    $P_k^{[n],[j]} = P_{k|k-1}^{[n],[j]}$
end for
Pre-compute measurement terms
for $j = 1...J_{k-1}^{[n]}$ do
    $\xi^{[n],[j]} = \text{measurement\_model}(\mu_{k|k-1}^{[n],[j]}, \vec{x}_k^{[n]})$
    $\mathrm{S}^{[n],[j]} = \mathrm{H}_k^{[n]} * P_{k|k-1}^{[n],[j]} * \text{transpose}(\mathrm{H}_k^{[n]}) + R$
    $\mathrm{K}^{[n],[j]} = P_{k|k-1}^{[n],[j]} * \text{transpose}(\mathrm{H}_k^{[n]}) * \text{pinv}(\mathrm{S}^{[n],[j]})$
    $P_k^{[n],[j]} = [I - \mathrm{K}^{[n],[j]} * \mathrm{H}_k^{[n]}] * P_{k|k-1}^{[n],[j]}$
end for
Update PHD component
$l = 0$
for $zz = 1...|Z_k|$ do
    $l = l + 1$
    $\alpha = 0$
    for $j = 1...J_{k-1}^{[n]}$ do
        $\eta_k^{[n],[l*J_{k-1}^{[n]}+j]} = P_d * \eta^{[n],[j]} * \text{mvnpdf}(z^{[zz]}; \xi^{[n],[j]}, \mathrm{S}^{[n],[j]})$
        $\alpha = \alpha + \eta_k^{[n],[l*J_{k-1}^{[n]}+j]}$
        $\mu_k^{[n],[l*J_{k-1}^{[n]}+j]} = \mu_{k|k-1}^{[n],[j]} + \mathrm{K}^{[n],[j]} * (z^{[zz]} - \xi^{[n],[j]})$
        $P_k^{[n],[l*J_{k-1}^{[n]}+j]} = P_k^{[n],[j]}$
    end for
    for $j = 1...J_{k-1}^{[n]}$ do
        $\eta_k^{[n],[l*J_{k-1}^{[n]}+j]} = \frac{\eta_k^{[n],[l*J_{k-1}^{[n]}+j]}}{\kappa+\alpha}$
    end for
    $J_k^{[n]} = l * J_{k-1}^{[n]} + J_{k-1}^{[n]}$
end for
```

Algorithm 4 Single cluster likelihood

```
$m_{k-1}^{[n]} = \sum_{j=1}^{J^{[n]}} \eta^{[n],[j]}$
$\gamma = 1$
for $zz = 1...|Z_k|$ do
    $\lambda = 0$
    for $j = 1...|J_{k-1}^{[n]}|$ do
        $\lambda = \lambda + \eta^{[n],[j]} * \text{mvnpdf}(z^{[zz]}; \xi^{[n],[j]}, \mathrm{S}^{[n],[j]})$
    end for
    $\gamma = \gamma * (\kappa + P_d * \lambda)$
end for
$w_k^{[n]} = \exp(m_{k-1}^{[n]}) * \gamma * w_{k-1}^{[n]}$
```

Algorithm 5 Resample

```
Given: $w_k^{[n]}$ and $N$ number of particles
Normalize: $w_k^{[n]} = \frac{w_k^{[n]}}{\sum_{j=1}^{J_k^{[n]}} w_k^{[n],[i]}}$
Calculate number of effective particle: $N_{eff} = \frac{1}{\sum_{j=1}^{J_k^{[n]}} (w_k^{[n],[i]} * w_k^{[n],[i]})}$
if $N_{eff} < T_{eff}$ then
    $r = \text{randn}(1)/N$
    $i = 1$
    $c = w_k^{[1]}$
    for $m = 1...N$ do
        $u = r + (m-1)/N$
        while $u > c$ do
            $i = i + 1$
            if $i > L$ then
                $i = L$
                break;
            end if
            $c = c + w_k^{[i]}$
        end while
        $w_k^{*[m]} = w_k^{[i]}$
        $\vec{x}_k^{*[m]} = \vec{x}_k^{[i]}$
        $P_k^{*[m]} = P_k^{[i]}$
        $\mathcal{M}_k^{*[m]} = \mathcal{M}_k^{[i]}$
    end for
else
    No resampling needed
end if
```