OpenPose picture

Bård-Kristian Krohg
baardkrk@student.matnat.uio.no

February 14, 2018

1 Placement Algorithm

1. Arrange Best matching limbs based on the score of the two connecting keypoints.
2. Select the N best matching limbs and calculate a scale S based on them.
3. Create vectors for all limbs under a certain threshold T.
4. Shift keypoints under a certian threshold T along the beam created by the 2D keypoint and the camera center. (increase Z and recalculate X and Y)
5. If the depth information at a keypoint is very uncertain, we should also rely on the depth information along the limb, flowing from the most certain keypoint to the least certain keypoint. This will also give us a vector along which we can place the point.
6. Calculate the difference in size between the observed limbs $\mathbf{L}_v$ and the model limbs $\mathbf{M}_v$[3]. The error E is then given by

$$E = \sum_{i=0}^{K} |\mathbf{L}_v(i) \cdot s(i)^3 - \mathbf{M}_v(i)|$$

Limb scores
When doing this it is important that we also check how similar the values are to each other. If they have a large gap, but high average, we don't want them amog the high scoring results.

$$\frac{a+b}{2} \cdot \lfloor a, b \rfloor^3 \quad | \quad a, b \in [0, 1]$$

Scale (height) If we have the vector of limb scores S and the vector of limb lengths L, desired limb lengths D we get the scale/height w by taking the weighted averages of each observed limb.

$$w = \frac{\sum_{i=0}^{n} \frac{L_i}{D_i} S_i}{\sum_{i=0}^{n} S_i}$$

[2] http://openlab.psu.edu/tools/proportionality_constants.htm
[3] change the name of these variables

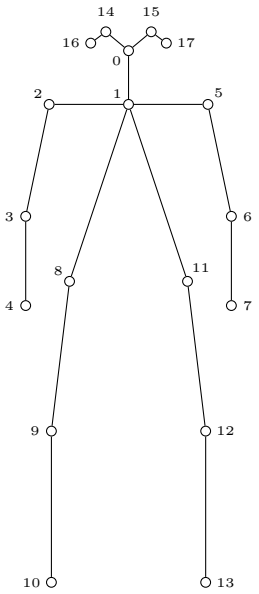


Figure 1: Numbering for OpenPose's keypoint markers.

ID	Name
0	Nose
1	Neck
2	Right Shoulder
3	Right Elbow
4	Right Wrist
5	Left Shoulder
6	Left Elbow
7	Left Wrist
8	Right Hip
9	Right Knee
10	Right Ankle
11	Left Hip
12	Left Knee
13	Left Ankle
14	Right Eye
15	Left Eye
16	Right Ear
17	Left Ear

Table 1: IDs for OpenPose's keypoint markers.

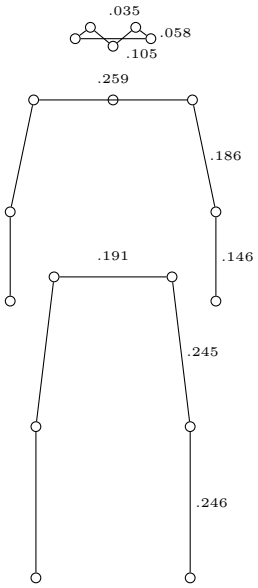


Figure 2: Constrained lengths[2].

ID	Limb	Model	Pair	$\pm t$
0	Left arm	.186	5 – 6	.05
1	Left forearm	.146	6 – 7	.05
2	Right arm	.186	2 – 3	.05
3	Right forearm	.146	3 – 4	.05
4	Hip	.191	8 – 11	.06
5	Left thigh	.245	11 – 12	.07
6	Left leg	.246	12 – 13	.07
7	Right thigh	.245	8 – 9	.07
8	Right leg	.246	9 – 10	.07

Table 2: Constrain rules for correct anthropometry. (Note that the anthropometry for the head and shoulders are not yet implemented.)

Doing it simple.

For each limb under the *threshold*:
Use S to get the scale of what the current limb's length *should* be.
create the "vector" of the limb to get the 3D direction in which to move/place the keypoint.
→ Creating the vector, search along the limb in the depth image. If we get to an edge, use the previous points to create the 3D vector (line) using RANSAC.

The limbs are stored as tuples: {`<score>, <length>`}

Creating the model parameters

For each person we need a set of model parameters to predict the 3D location of each keypoint. These parameters consist of $\mathbf{t}$ – a set of numbers representing the leeway we give to each limb in relation to the model. The formula we get for fitting the human model to the observed locations is

$$Error = \sum_{alllimbs} (scale(model\_length + \mathbf{t}_i) - actual\_length) \cdot score\_of\_observed\_keypoint$$

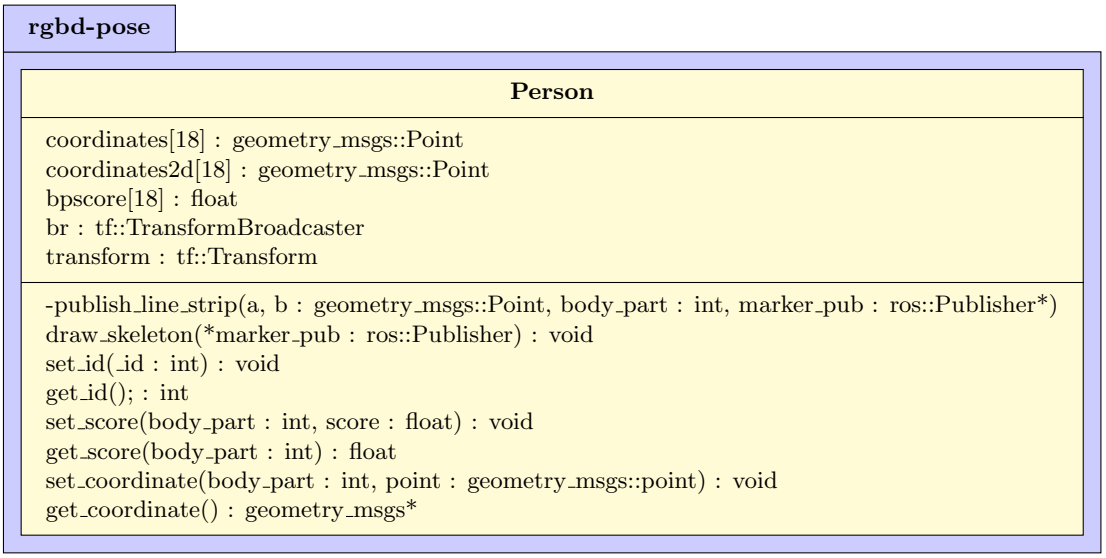
$$E = \left[S \begin{bmatrix} ML + \mathbf{t}_i \\ \vdots \end{bmatrix} - \begin{bmatrix} ac\_L \\ \vdots \end{bmatrix} \right]^T \cdot \begin{bmatrix} score \\ \vdots \end{bmatrix}$$

where we minimize the error by changing $\mathbf{r}$ and *scale*. If we already have calculated a previous set of parameters we should take this previous model into consideration. Therefore each model should be given a score, based on how "old" it is, the score of the parameters used in creating the model (the keypoint score for the N limbs used to create the model). The "old" part can be fixed by increasing the score after each iteration.

Each person should therefore have their own variables with $\mathbf{t}$, *scale* and *current_score*.

After we now have the predicted lengths for each limb, we can estimate the 3D positions for the different keypoints. Here, we should also have the previous or predicted keypoints (Kalman filter) to smooth the movement of the person. This should also be weighted however, because an observation with high score should trump a prediction any time. However, if we have no observation, we can use the predicted position from the previous frame to also track non-observed limbs. We should also include a score for this, so we know which positions have had the most modification by this algorithm. This can be an input to the activity recognition algorithm to possibly account for wrong predictions and other noise.

2 The Program UML



3 rgbd-pose

3.1 Person

Storage class for each person object. Every person has an *ID* and a set of *coordinates* (2d and 3d). Each person also has a *base_frame* which is a string. The person class is able to publish line strips between each of it's keypoints, and thus draw itself in rviz.

3.1.1 ID

is a unique identifier for each person, and is an integer for the moment.

3.1.2 Coordinates

we have two separate coordinate arrays, as one stores the 2D image positions of each keypoint, and the other stores the 3D calculated positions of each keypoint. The 3D coordinates are subject to change when we constrain the pose using the `pose_estimator` class.

3.1.3 base_frame

tells us what this persons coordinates are calculated from. We can then translate the persons coordinates to other ROS frames, as long as we know their relation to this *base_frame*.

3.2 pose_estimator

A utility class for gathering all functions that helps us estimate the 3D pose of the persons. This class also runs the openpose program on provided frames. We

create this object only one time, but it is updated with new *depth frames* and *frames*. This is because OpenPose runs on individual frames, and not on video sequences. In addition the `pose_estimator` class has parameters for the *focal length* and the *pixel center* of the camera model for the frames. We need this when we place the points into 3D space.

4 Notes

Time sync for ROS-TMS: do the same as in "Multiple Depth Cameras Calibration and Body Volume Reconstruction for Gait Analysis". (Separate computers are synched with NTP, send information to central server. This runs.. wait there is a problem with subscribing to topics with ROS... ALTHOUGH, if the separate computers are good enough, they could calculate the 3D positions and publish q

some ideas for abstract: The easiest way to diagnose a patient is to go to them. .. many patients .. no time .. better with routine checks ..

Hjemmesykepleien sliter med ufaglærte. derfor bedre med et automatisk system som kan hjelpe til med å kjenne igjen de viktigste faresignalene hjemme. Et "alt i ett" system som kan passe på tabeletter osv i tillegg.

List of Figures

1 Numbering for OpenPose's keypoint markers 2
2 Constrained Lenghts 2

List of Tables

1 IDs for OpenPose keypoint markers 2
2 Anthropometry constrain rules 2

5 Solving multiple view stereo