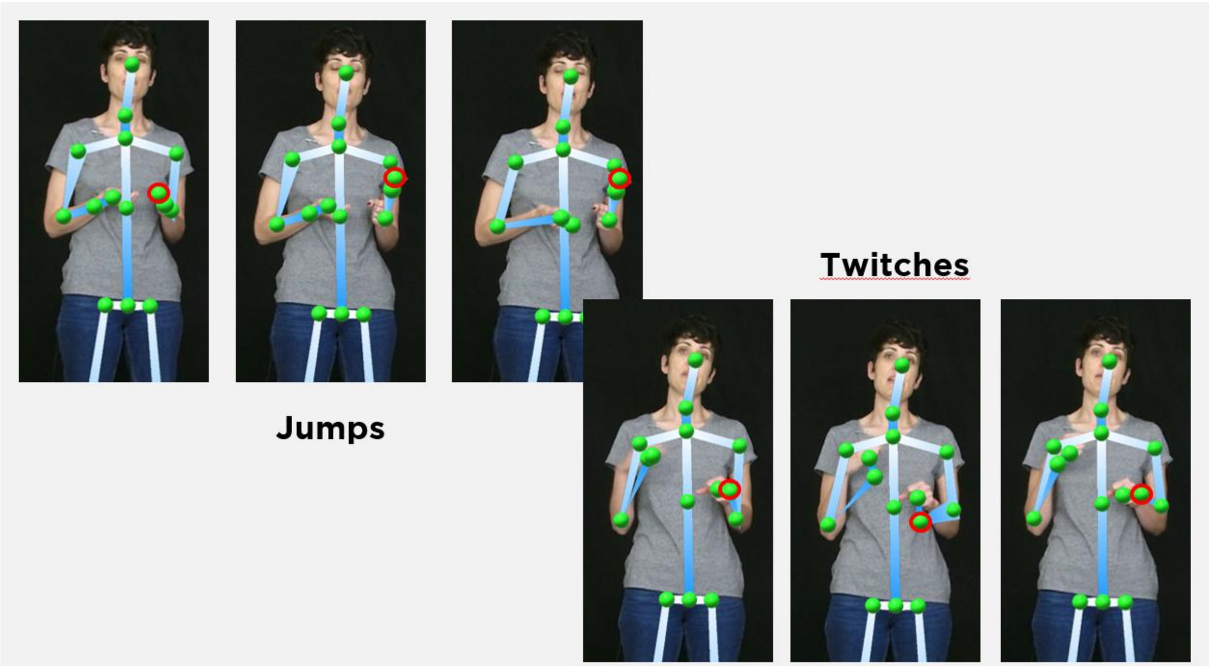
KRAJJAT

Introduction

KRAJJAT allows to smoothen the position of joints from a Kinect recording, to correct for the big **jumps** and **twitches** that can appear between two frames, noticeably on the hands and arms.

- **Jumps** are defined as unrealistic displacements of the position of a joint in between frames, **without** a subsequent return at the original position under a few frames. In the example below, the right hand joint suddenly moves between frames 1 and 2, and remains in the position in frame 3.
- **Twitches** are defined as unrealistic displacements of the position of a joint in between frames, **with** a subsequent return at the original position under a few frames. For instance, in the image below, the right wrist joint suddenly moves between frames 1 and 2, but comes back to the original position on frame 3.



How the realignment works

KRAJJAT works simply by defining two variables:

- The **threshold**, being the amount of movement between two frames over with we consider a displacement to be unrealistic. It is defined in centimetres per second.

For every joint in every frame, a displacement is calculated, in centimetres per second. To do so, a distance is first calculated between the position of the joint at two subsequent frames. Then, this distance is divided by the elapsed time between the two frames. The result, expressed in centimetres per second, is **compared to the threshold.** If the result is over to the threshold, then the program will correct the position of the joints depending on if they are a twitch or a jump. The difference between the two depends on the number of frames.

- The **window** is the amount of frames defining if an over-threshold displacement is a jump or a twitch.

For each frame of the window, the program checks if the position of the joint comes back in the limits of a sphere:

- Centered around the position of the joint before the first frame of the window
- Of a radius equating the threshold

If the position of the joint does come back in the bounds of the sphere after a number of frames inferior to the length of the window, the algorithm considers the displacement as a twitch, and defines all of the frames where the joint is out-of-bounds as being in the interval of correction.

The program then considers a line between the positions of the joint at the frame before the interval and after the interval, and replaces every joint from the interval at a specific fraction of this line – the fraction depending on the time stamp of every frame of the interval.

Finally, every corrected window is marked as "corrected" to avoid multiple corrections in the case of subsequent twitches.

Note that if the joint position does not come back in the bounds of the sphere after reaching the maximal amount of frames for a window, it performs a correction on the frames before in the same way, in order to smoothen out the jump. The only difference with a twitch is that the joint position after the interval is out of the bounds of the sphere.

Realignment algorithm

Let's call j_{t_p} a joint, as being a 3-dimensional (x, y, and z, in cm to the origin) point being located in space at a moment t_p (in seconds), where p iterates through all the poses of a video. For a speed threshold s and a window w, if the following inequation is true:

$$(\mathbf{1})\ \frac{\sqrt{\left(x_{j_{t_{p+i+1}}} - x_{j_{t_{p+i}}}\right)^2 + \left(y_{j_{t_{p+i+1}}} - y_{j_{t_{p+i}}}\right)^2 + \left(z_{j_{t_{p+i+1}}} - z_{j_{t_{p+i}}}\right)^2}}{t_{p+i+1} - t_{p+i}} \geq s \text{ for } i \in [0, w] \text{ and } i \in \mathbb{N}$$

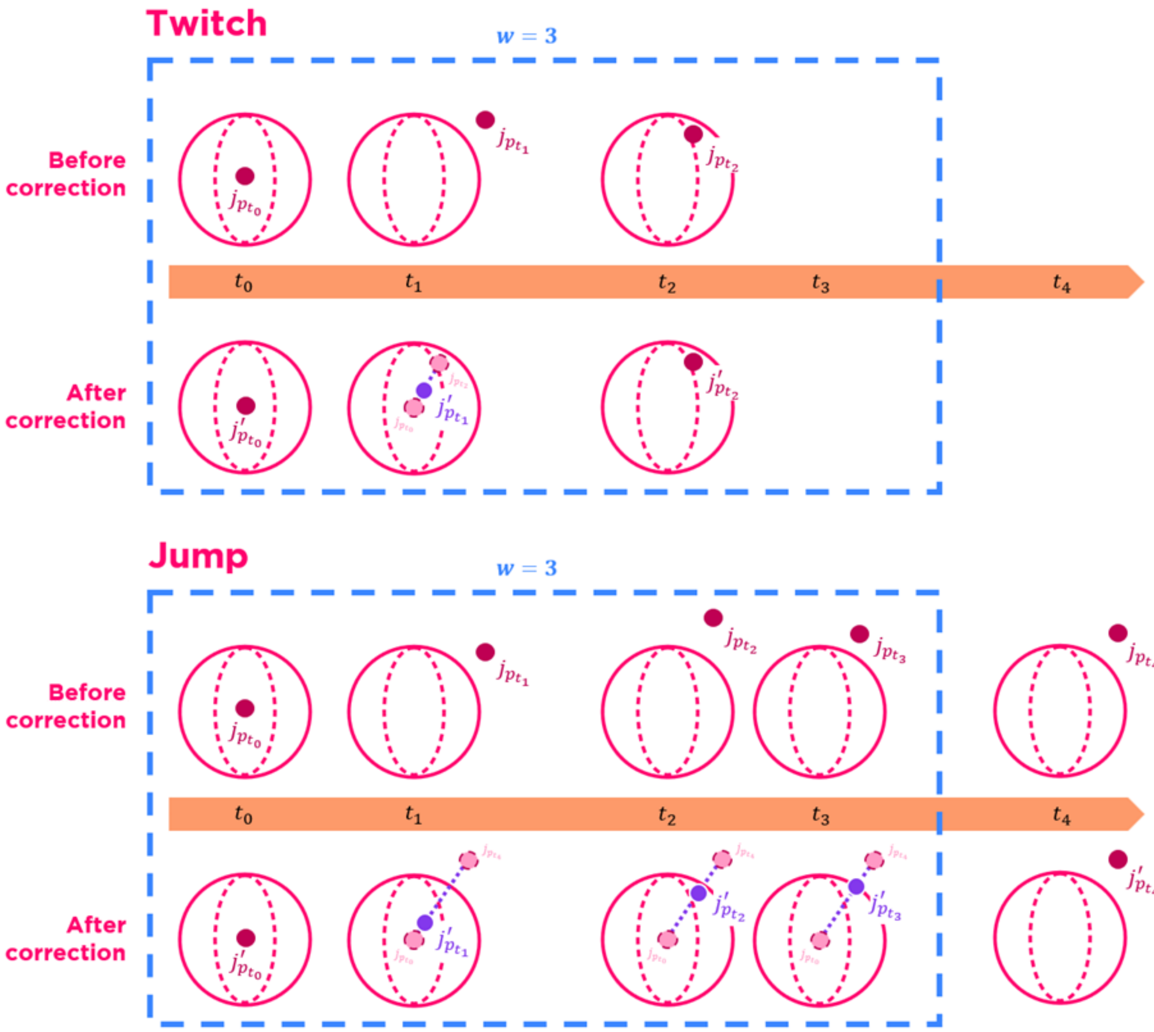
the misplacement is considered as a **jump**. If for any $i \in [0, w]$, the inequation $(\mathbf{1})$ is false, the misplacement is considered as a **twitch**.

Once the misplacement has been properly labelled, the realignment can be performed. For every joint $j_{t_{p+i}}$ with $i \in [1, v[$ and $i \in \mathbb{N}$, with v being equal:

- To the length of the window w in the case of a **jump**;
- To a number $i \in [1, w]$ being the first number for which the inequation $(\mathbf{1})$ is false, in the case of a **twitch**;

The coordinates are realigned such as:

$$
(2)\begin{cases}
x'_{j_{t_{p+i}}} = x_{j_p} - \dfrac{t_{p+i} - t_p}{t_{p+v} - t_{p+i}} \times \left(x_{j_p} - x_{j_{p+v}}\right) \\
y'_{j_{t_{p+i}}} = y_{j_p} - \dfrac{t_{p+i} - t_p}{t_{p+v} - t_{p+i}} \times \left(y_{j_p} - y_{j_{p+v}}\right) \\
z'_{j_{t_{p+i}}} = z_{j_p} - \dfrac{t_{p+i} - t_p}{t_{p+v} - t_{p+i}} \times \left(z_{j_p} - z_{j_{p+v}}\right)
\end{cases}
$$



Schema showing the principle of a realignment of a twitch (top) and of a jump (bottom). In our example, $w = 3$. Note that the variable interval of time between two frames/poses has been exaggerated for demonstration. On top of the orange timeline for each schema is the original position of one joint, before realignment; on the bottom of the timeline is the position of the joint after realignment. For both the twitch and the jump, on t_1, the joint $j_{p_{t_1}}$ is detected outside the threshold sphere (of radius s) traced around the joint $j_{p_{t_0}}$. A realignment is then necessary. For the twitch, on t_2, the joint position comes back within the threshold circle traced around the joint $j_{p_{t_0}}$: the window being equal to 3, the misplacement is labelled as a twitch. Then, a line is traced between the joints $j_{p_{t_0}}$ and $j_{p_{t_2}}$ (here, in purple). The new point $x'_{j_{t_1}}$ is positioned on this line, with its relative distance between $j_{p_{t_0}}$ and $j_{p_{t_2}}$ equal to the relative position of t_1 between t_0 and t_2. For the jump, the joint remains outside of the threshold sphere for a number of poses superior to the window: the misplacement is labelled as a jump. The same realignment as the one described for the twitch is performed, with the exception that the ending point of the realignment is outside of the threshold sphere.