684,000 people die from falls globally eve

the **second leading cause of unintentional injury**

A **1-Hour Delay** in medical assistance following f

doubles the mortality rate in the following six m

Fall Pose Estimation Detectior

Calculation metrics

Spatial & Pose Structure Metrics

Bounding Box Aspect Ratio (AR) determines if the box containing the person is "wide" (horizontal) or "tall" (vertical).

$$AR = \frac{x_{max} - x_{min}}{\max(1, y_{max} - y_{min})}$$

$$\text{mid}_{shoulder} = \frac{P_{L.sh} + P_{R.sh}}{2}, \quad \text{mid}_{hip} = \frac{P_{L.hip} + P_{R.h}}{2}$$

Torso Angle (θ). Calculated by finding the angle between the *mid-shoulder point and the mid-knee point* relative to the vertical axis.

$$\theta = \arctan 2(|\Delta x|, |\Delta y|) \times \frac{180}{\pi}$$

$$\Delta x = M_{knee}(x) - M_{shoulder}(x)$$

Anatomical Ruler (Length of Torso). The "ruler" is the distance between the *shoulders and hips*, used to normalize movement regardless of how close the person is to the camera.

$$L_{torso} = \text{median}(\{L_{inst,1}, L_{inst,2}, .$$

$$L_{inst} = \sqrt{(x_{sh} - x_{hip})^2 + (y_{sh} - y_h}$$

#1: The Torso Ruler (Velocity Nor

Standard computer vision measures speed in pixels. This is unreliable because a person *close* to the camera (**Fig. 2a**) appears to move "faster" than someone *far away* (**Fig. 2b**).

My algorithm calculates the *median torso length*, or the *distance between shoulders and hips* (**Fig. 1**). The system measures movement relative to the user's own body size, making it distance-invariant. During fast descent or when the person lies down, it freezes to maintain a normal value and updates when the person stands up.

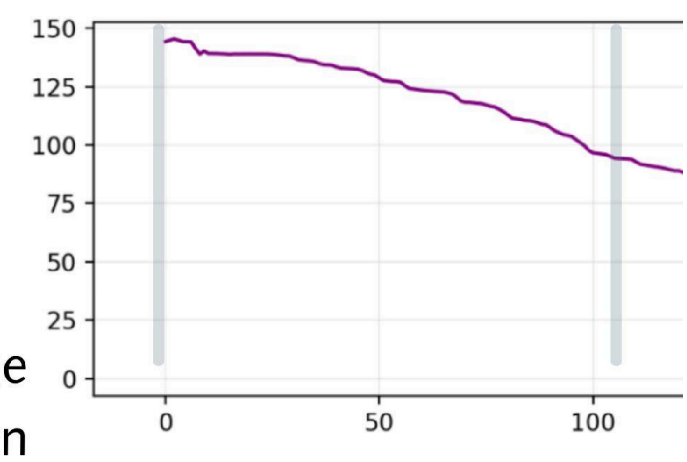
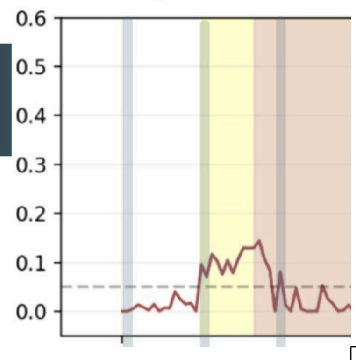


Figure 1: Graph of ruler from **Figure 2a** (frame 0, 146 pixels) to **Figure 2b** (frame 110, 83 pixels)

#2: Laying Down Posture Analysis



Once a rapid descent (0.0625 torso lengths change in one frame) is detected, the program calculates the angle of the vector from the shoulders to the knees. If this angle exceeds 35°, an initial fall

very year,

ury deaths

g falls

months

Seniors in households with annual incomes under **$20,000** have a **4%** higher prevalence of fall injuries, where expensive medical monitoring and fall-detection wearables are unavailable or too expensive

on Algorithm

s

Motion Metrics

Normalized Vertical Velocity: Determines the downward speed of the body center, scaled by the person's own torso size to be distance-invariant.

$$V_y = \frac{\bar{y}_t - \bar{y}_{t-1}}{L_{torso}}$$

$P_{R.hip}$

) $\Delta y = M_{knee}(y) - M_{shoulder}(y)$

$_{t,2}, \ldots, L_{inst,30}\})$

$- y_{hip})^2}$ $\bar{y}_t = \frac{y_{L.sh} + y_{R.sh} + y_{L.hip} + y_{R.hip}}{4}$

ormalization)

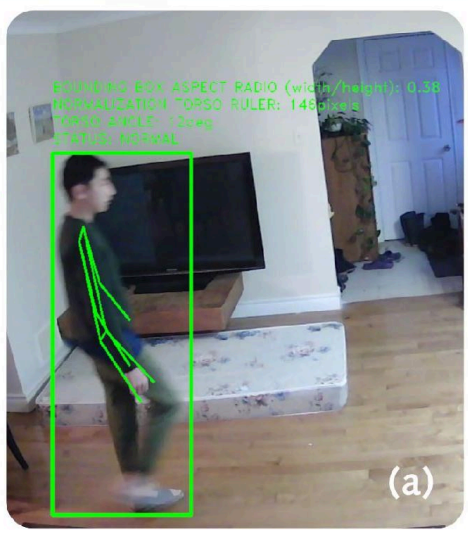
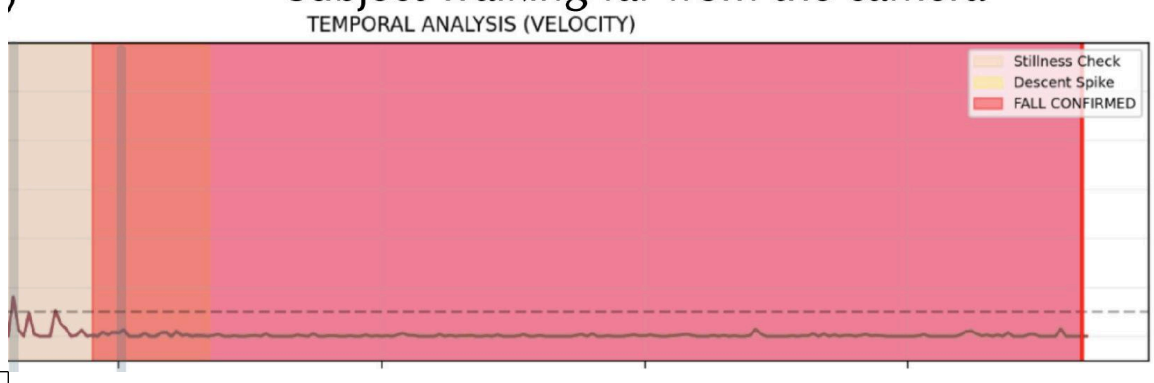
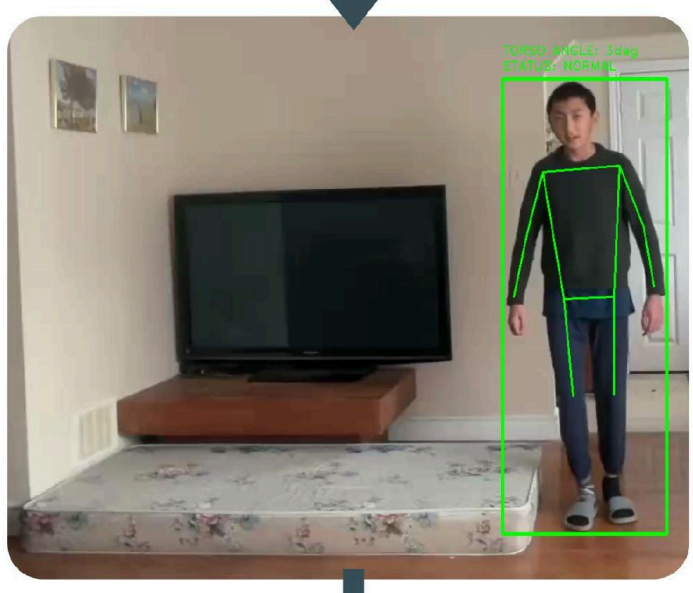


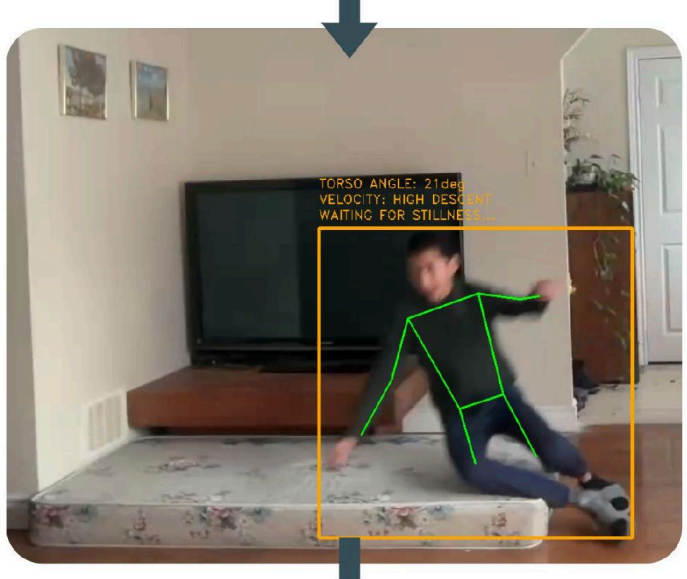
Figure 2: Comparison of perspective distortion in movement. **(a)** Subject walking near the camera. **(b)** Subject walking far from the camera

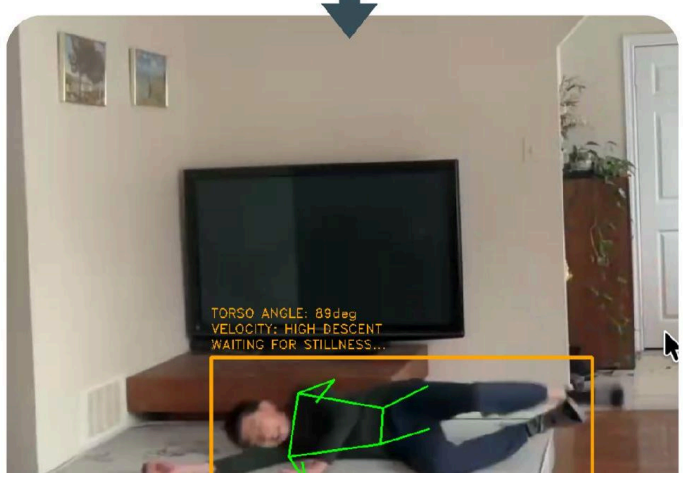


FallCam Device (Pi Zero 2W, Thinkpad, ESP32 CAM, etc...)

Tailscale Encrypted Tunnel VPN → Mac Dashboard







the shoulders to the knees. If this angle exceeds 35°, an initial fall is flagged.

In addition, to catch forward falls where the skeletal angle might look vertical from the camera's perspective, I integrated a bounding box aspect ratio check as backup (width/height >2).

#3: The Confirmation Window

Most vision-based systems trigger an alert based on a single frame, which leads to "False Positives" from many daily activities. To ensure a fall is a crisis and not a stumble, the system initiates a **lay-down stillness timer (Fig. 3).** The algorithm requires 30 consecutive frames (approx. 1 second) of near-zero velocity while in a horizontal state. I also added a coordinate check: the subject must be in the lower 45% of the frame (Y > 0.55) to confirm they are on the floor. This confirmation window effectively filters out common daily movements.

Dealing with Data Occlusion and Gaps

Falling behind furniture (chairs, beds) causes data gaps. To solve it, I Implemented a Temporal Memory Buffer that maintains an alert state during visual gaps, ensuring the system confirms the fall once the subject is partially re-visible.

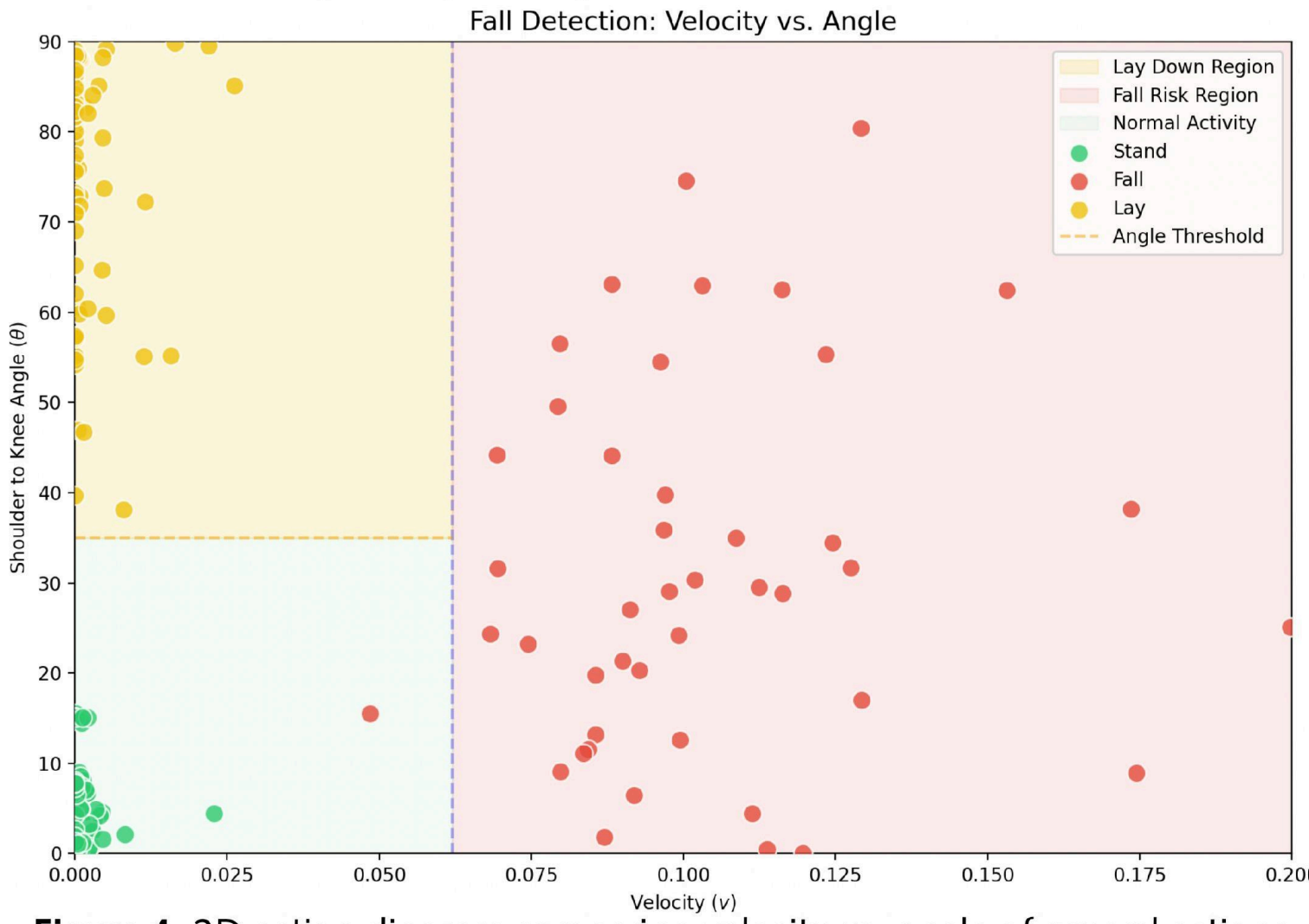


Figure 4: 2D action diagram comparing velocity vs. angle of several actions

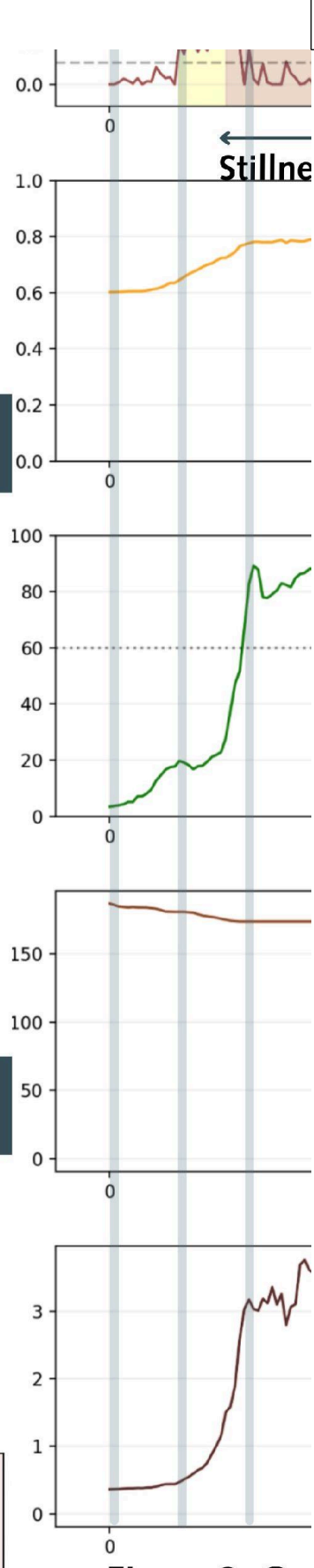
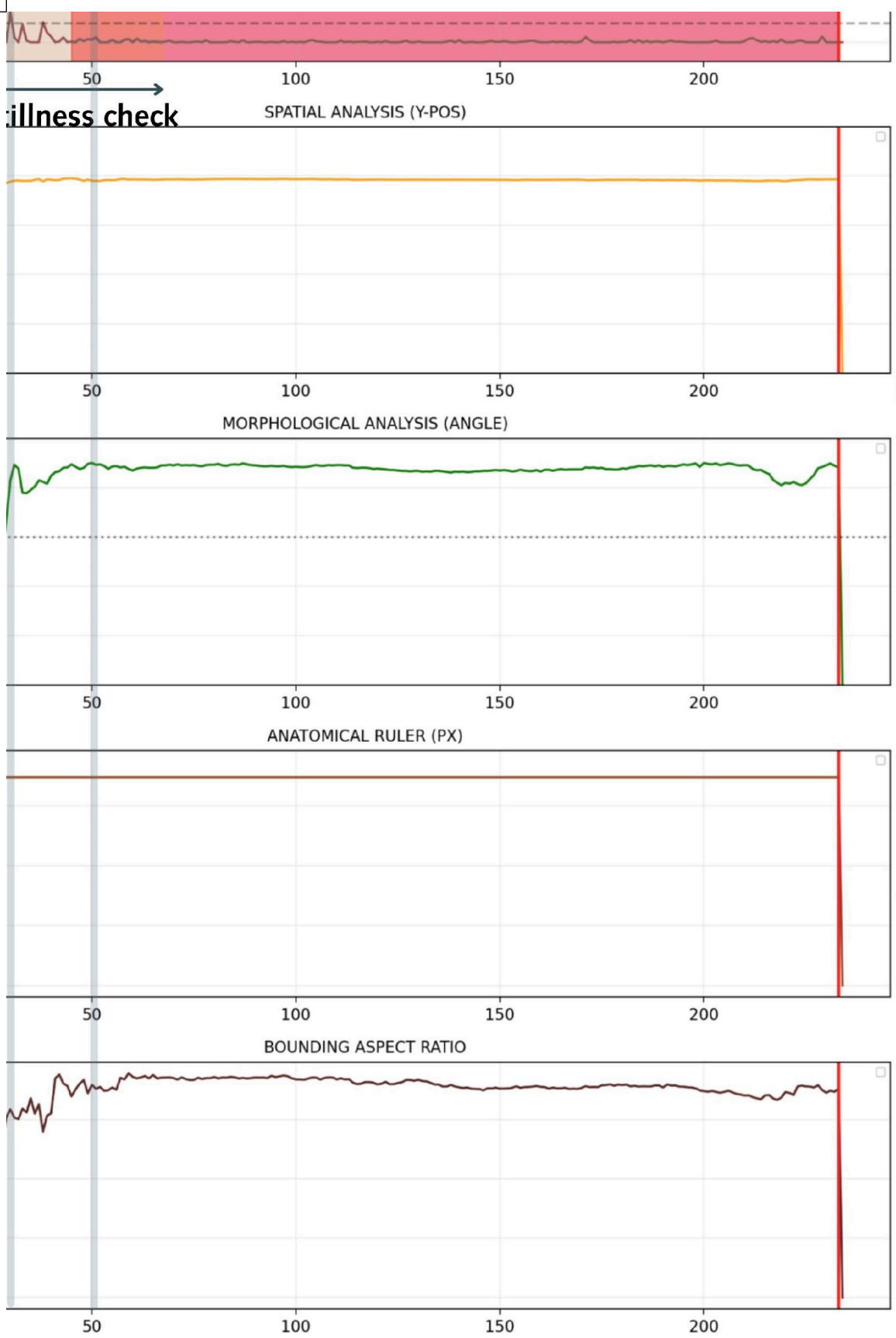


Figure 3: Gra
normalization ru
laying d

By plotting th
in 3D (velocit
vs. y-position
2D (velocity
4), we can se
'Stand' and 'l
form tight, pr
clusters (**Fig.**



: Graph overview of the velocity, y-position, body angle, on ruler, and aspect ratio. Fall detection stages (fast descent, ng down, confirmation window, fall) are highlighted

g the action data ocity vs. angle tion, **Fig. 5**) and ity vs. angle, **Fig.** n see that nd 'Lay' events t, predictable **ig. 5).**

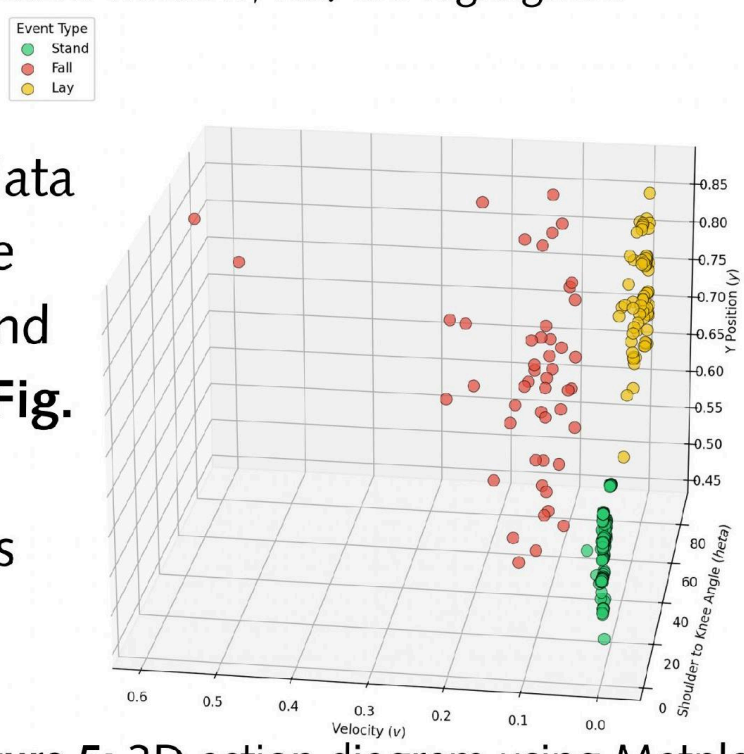


Figure 5: 3D action diagram using Matplotlib