Free-hand point-cloud gesture recognizer

Elena-Gina Craciun  
University Stefan cel Mare of Suceava  
Suceava 720229, Romania  
ginac@eed.usv.ro

Ionela Rusu  
University Stefan cel Mare of Suceava  
Suceava 720229, Romania  /ionelar@eed.usv.ro

Radu-Daniel Vatavu  
University Stefan cel Mare of Suceava  
Suceava 720229, Romania  
vatavu@eed.usv.ro

Our free-hand recognizer extends the $P^+$ gesture recognizer to 3-D gestures performed by the hand in mid-air. In the following pseudocode, HandPose is a structure that exposes $x$ and $y$ coordinates for each finger of the hand, such as delivered by the Leap Motion controller. The pseudocode follows the main structure of the $P^+$ recognizer.

---

**FreeHandRecognizer** (List HandPose, C, Templates templates)

1: n ← 32
2: Normalize(C, n)
3: score ← ∞
4: for each T in templates do
5: Normalize(T, n) // should be pre-processed
6: d ← min(Cloud-Distance(C, T), Cloud-Distance(T, C))
7: if score > d then
8: score ← d
9: result ← T
10: return (result, score)

---

**Cloud-Distance** (List HandPose, C, List HandPose T, int n)

1: matched ← new bool[n]
2: sum ← 0
3: // match hand poses from cloud C with poses from T; 1-to-many matchings allowed
4: for i = 1 to n do
5: min ← ∞
6: for j = 0 to n do
7: d ← HandPose-Distance(Ci, Tj)
8: if d < min then
9: min ← d
10: index ← j
11: matched[index] ← true
12: sum ← sum + min
13: // match remaining hand poses T with poses from C; 1-to-many matchings allowed
14: for j such that not matched[j] do
15: min ← ∞
16: for i = 1 to n do
17: d ← HandPose-Distance(Ci, Tj)
18: if d < min then min ← d
19: sum ← sum + min
20: return sum

---

**Gesture normalization**. Gesture points are resampled, scaled with shape preservation, and translated to origin.

Normalize (List HandPose gesture, int n)

1: gesture ← Resample(gesture, n)
2: Scale(gesture)
3: Translate-to-Origin(gesture, n)

---

**Gesture rescaling**. Rescale gesture with shape preservation so that the resulting bounding box will be ∋ [0..1] × [0..1] × [0..1].

Scale (List HandPose gesture)

1: xmin ← ∞; xmax ← 0; ymin ← ∞; ymax ← 0
2: for each hand in gesture do
3: for i = 1 to hand.numFingers do
4: xmin ← Min(xmin, hand.x[i])
5: ymin ← Min(ymin, hand.y[i])
6: xmax ← Max(xmax, hand.x[i])
7: ymax ← Max(ymax, hand.y[i])
8: scale ← ∆xmax / ∆xmin = ∆ymax / ∆ymin
9: for each hand in gesture do
10: for i = 1 to hand.numFingers do
11: hand.x[i] ← (hand.x[i] - xmin) / scale
12: hand.y[i] ← (hand.y[i] - ymin) / scale

---

**Hand pose distance**. Computes the distance between two hand poses as the sum of Euclidean distances between their fingers’ coordinates.

HandPose-Distance (HandPose a, HandPose b)

1: d ← 0
2: for i = 1 to a.numFingers do
3: d ← d + ∥ai - bi∥
4: return d

---

Project no. PN-II-RU-TE-2014-4-1187, financed by UEFISCDI, Romania.