APPENDIX A: PSEUDOCODE

We present in this section minimal pseudocode for $P+ based on the $P pseudocode from Vatavu *et al.* [46] (p. 280). For space concerns, we only list the parts of $P that we updated. Complete pseudocode for $P+ as well as C# and JavaScript implementations are available at `http://www.eed.usv.ro/~vatavu`. In the following, Point is a structure that exposes x and y coordinates, the stroke ID, and the normalized turning angle θ. Points is a list of points and Templates a list of Points with gesture class data.

```
$P+Recognizer (Points $C$, Templates $templates$)
$n \leftarrow 32$
Normalize($C$, n)
$score \leftarrow \infty$
for all $T$ in $templates$ do
    Normalize($T$, n) // should be pre-processed
    $d \leftarrow \min\left(\text{Cloud-Distance}(C,T), \text{Cloud-Distance}(T,C)\right)$
    if $score > d$ then
        $score \leftarrow d$
        $result \leftarrow T$
return $\langle result, score \rangle$
```

```
Cloud-Distance (Points $C$, Points $T$, int $n$)
$matched \leftarrow$ new bool[$n$]
$sum \leftarrow 0$
// match points from cloud $C$ with points from $T$; one-to-many matchings allowed
for $i = 1$ to $n$ do
    $min \leftarrow \infty$
    for $j = 1$ to $n$ do
        $d \leftarrow$ Point-Distance($C_i$, $T_j$)
        if $d < min$ then
            $min \leftarrow d$
            $index \leftarrow j$
    $matched[index] \leftarrow true$
    $sum \leftarrow sum + min$
// match remaining points $T$ with points from $C$; one-to-many matchings allowed
for all $j$ such that not $matched[j]$ do
    $min \leftarrow \infty$
    for $i = 1$ to $n$ do
        $d \leftarrow$ Point-Distance($C_i$, $T_j$)
        if $d < min$ then $min \leftarrow d$
    $sum \leftarrow sum + min$
return $sum$
```

```
Point-Distance (Point $a$, Point $b$)
return $\left((a.x - b.x)^2 + (a.y - b.y)^2 + (a.\theta - b.\theta)^2\right)^{\frac{1}{2}}$
```

```
Normalize (Points $points$, int $n$)
$points \leftarrow$ Resample($points$, $n$)
Scale($points$)
Translate-to-Origin($points$, $n$)
Compute-Normalized-Turning-Angles($points$, n)
```

```
Compute-Normalized-Turning-Angles (Point $C$, int $n$)
$C_{1.\theta} \leftarrow 0, C_{n.\theta} \leftarrow 0$
for $i = 2$ to $n - 1$ do
    $C_{i.\theta} \leftarrow \frac{1}{\pi} \arccos\left(\frac{(C_{i+1.x} - C_{i.x}) \cdot (C_{i.x} - C_{i-1.x}) + (C_{i+1.y} - C_{i.y}) \cdot (C_{i.y} - C_{i-1.y})}{\|C_{i+1} - C_i\| \cdot \|C_i - C_{i-1}\|}\right)$
return
```