$N$-Protractor Pseudocode

Lisa Anthony‡ and Jacob O. Wobbrock‡

†University of Maryland Baltimore County, ‡The Information School, DUB Group, University of Washington

The Protractor-enhanced $N$ pseudocode is presented here to aid implementation. In addition, the $N$ webpage includes open-source JavaScript and C# implementations. Operations that have not changed since the original $N$ are noted; for those details, see [1]. For results of the SN-Protractor evaluations, see [2].

**Step 1.** Take a multistroke gesture $\text{strokes}$ and generate unistroke permutations of it. For gestures serving as templates, Step 1, which uses Steps 3-7, should be carried out once on the input points. For candidates, Steps 2-8 should be applied to the input points. For constants we use $N=96$, $size=250$, $\varphi=30°$, $O=(0,0)$, and $I=12$. IEEE-PERMUTE and MAKE-UNISTROKES are unchanged from Step 1 in [1].

**Step 2.** Combine candidate strokes into one unistroke points path. COMBINE-STROKES remains unchanged from Step 2 in [1].

**Step 3.** Resample a points path into $n$ evenly spaced points. RESAMPLE remains unchanged from Step 3 in [4].

**Step 4.** Find and save the indicative angle $\omega$ from the points’ centroid to first point. Then rotate by $-\omega$ to set this angle to $0°$. INDICATIVE-ANGLE and ROTATE-BY remain unchanged from Step 4 in [1].

**Step 5.** Scale in a dimensionally-sensitive fashion based on threshold $\varphi=30°$. Next, if using bounded rotation invariance, restore drawn orientation by rotating $+\varphi$. Then translate to the origin $O=(0,0)$. SCALE-DIM-TO, TRANSLATE-TO and CHECK-RESTORE-ORIENTATION remain unchanged from Step 5 in [1].

**Step 6.** Calculate the start unit vector $\text{v}_{\text{start}}$ for points using index $I=12$. CALC-START-UNIT-VECTOR remains unchanged from Step 6 in [1].

**Step 7.** Create the vector version of points. This step has been changed for SN-Protractor from Protractor’s original formulation [3] to remove calls to RESAMPLE and TRANSLATE-TO, since SN-Protractor already performs these operations in Steps 3 and 5.

**Step 8.** Match vector version $\text{vector}$ of candidate points having start unit-vector $\text{v}_{\text{start}}$, processed from the raw strokes in Steps 2-7 prior to calling RECOGNIZE, where now $S = \{\text{strokes}\}$, against unistroke permutations $U$ within each multistroke $M$. We use $\Phi = 30°$ for the start angle similarity threshold. OPTIMAL-COSINE-DISTANCE is the matching method from Protractor [3].

**REFERENCES**


---

1 http://depts.washington.edu/aimgroup/proj/dollar/ndollar.html