Estimating the Pen Trajectories of Handwritten Static Scripts using Hidden Markov Models

Emli-Mari Nel

Department of Electrical and Electronic Engineering
University of Stellenbosch
South Africa

General presentation 2006

Outline

Dynamic/on-line handwriting recognition

- **Digitising tablet:**

- Capture pen dynamics:
  - position
  - velocity
  - tilt
  - pressure
- Expensive: Tablet at each signing post.

Static/off-line handwriting recognition

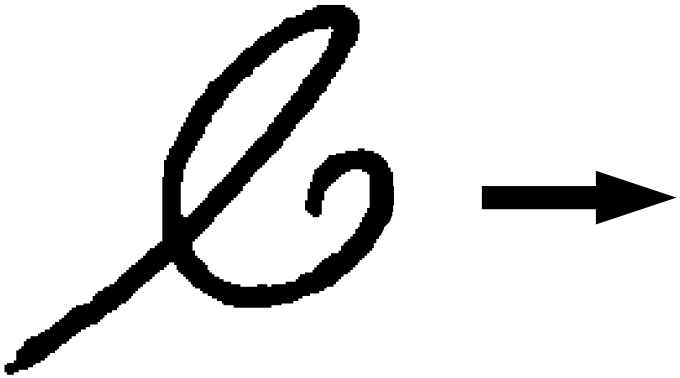
Scanner:

- Capture 2D images.
- Static systems: information loss.
- Static systems are less reliable.

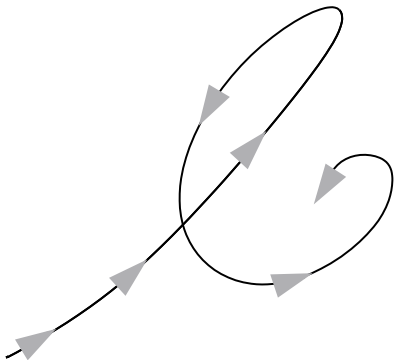
Pen trajectory estimation

- **Convert** static to dynamic.
- **Improve** the accuracy of static systems.
- Estimate **pen trajectories**.

Static script



Parametric curve



Applications

Off-line character recognition:

- Text retrieval from historical documents.
- Automatic processing of text on envelopes or cheques.
- Number plate recognition.

Applications

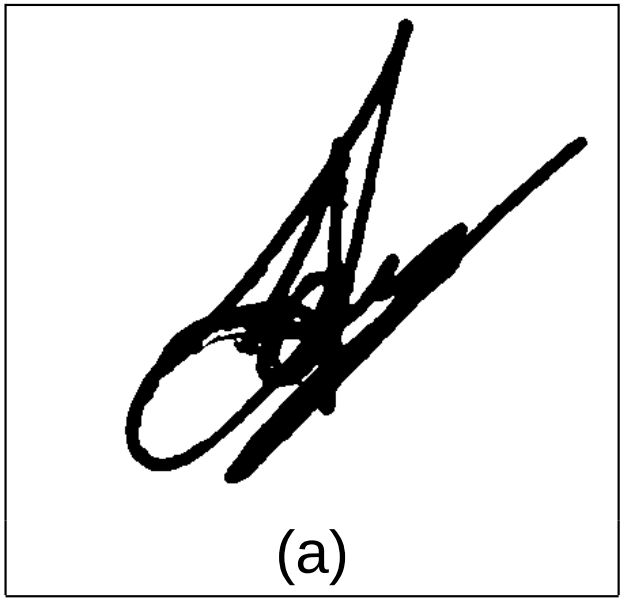
Off-line static signature verification:

- Cheques: Bank industry.
- Credit/debit card transactions.
- Other contractual documents.

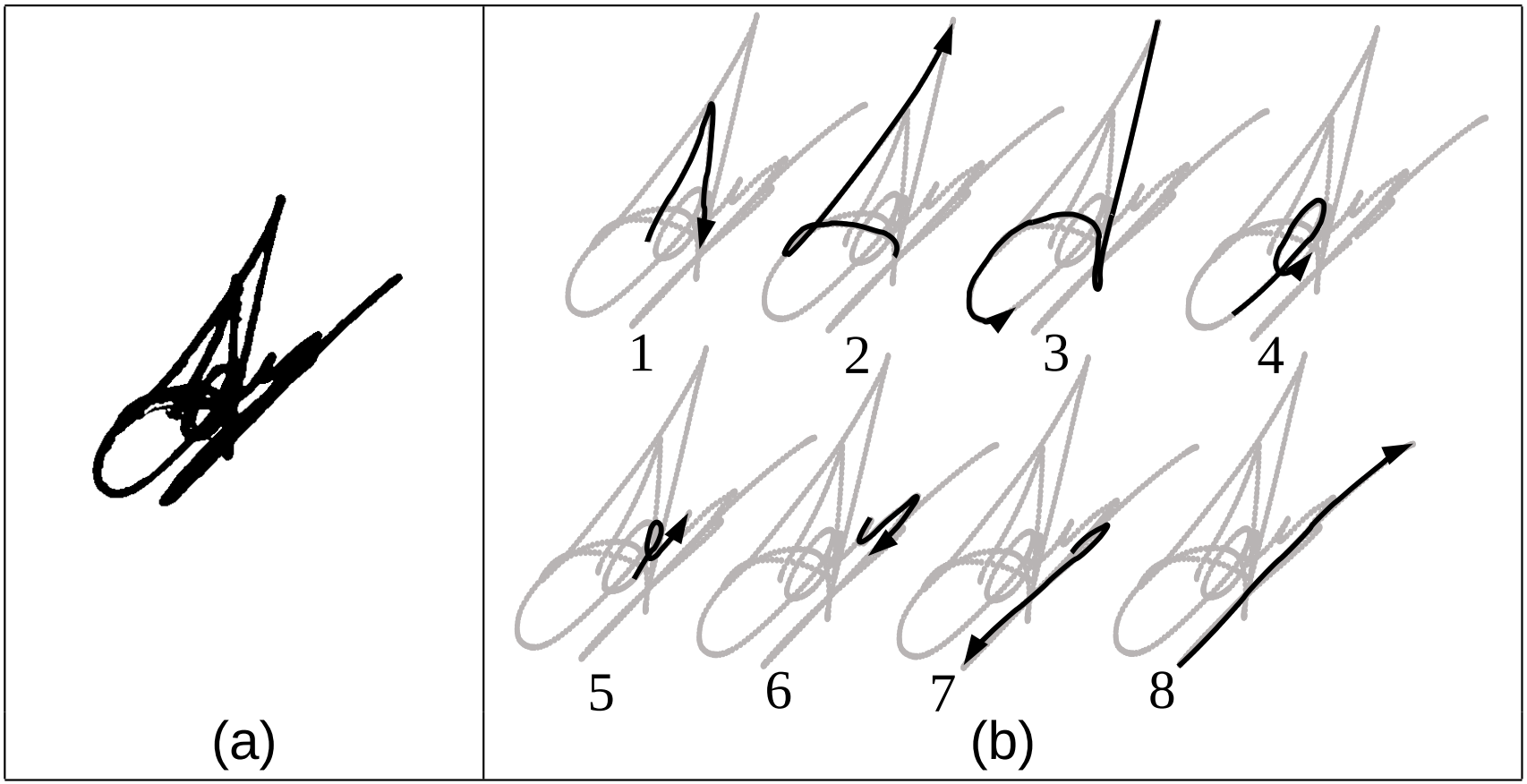
Problem statement: Subproblems

Subproblems:

1. Identify starting/terminating points.
2. Unravel intersections.
3. Identify turning points.
4. Identify pen-up/pen-down events.



Problem solved: available dynamic counterpart



Note: **Dynamic counterpart is not available!**

Existing approaches (1991 - 2004) 10

- Rule-based methods.
- Graph-theoretical methods.
- Local correspondence methods.

Rule-based methods

- Earlier approaches.
- Rules mimic underlying handwriting principles.
- Disadvantages:
  - Population-based rules,
  - e.g., Westerners write from left to right.
  - Local smoothness criterion.
  - Limit number of intersecting lines.

Graph-theoretical methods: Background

- Construct graph from static image.
- Assumption of motor-controlled pen motions:
  - Continuous handwriting motions.
  - Minimum energy.
- Define global cost function to minimise.
- Calculate shortest path.

Graph-theoretical methods: Background

- Mostly, find solution to:
  - Chinese postman problem.
  - Travelling salesman problem (TSP).
- Advantages:
  - Global optimisation.
  - Resolve ambiguities more accurately.

Graph-theoretical methods: Pros and Cons

Disadvantages:

- Computationally intensive.
- TSP: inefficient (NP complete.)
- Pen-up events → discontinuities.
- Invalidates motor-control continuity assumption.
- Assume single path.

Local correspondence approach

- Guo et al. (2001): Include prerecorded information.
- Single tablet: Record **different** dynamic representatives/exemplars.
- Establish point-wise correspondence.
- **Disadvantages**:
  - Cannot resolve ambiguities accurately.
  - Locally optimised search algorithm.

Development outline

Category

- Local correspondence methods.
- **Prior data:** Prerecorded **dynamic exemplars**.

Input

- Scanned 2D image of document.

Preprocessing

- Binarisation.
- **Thinning.**
- **Alignment:** Scale, translation, **rotational** invariance.
- Resampling.

Development outline

Statistical modelling of static script

- Derive a hidden Markov model **(HMM)**.
- **Match** dynamic exemplars to HMM.
- Use **known** sequences to unravel static image.

Output

- **Pen trajectory** of static script.
- Log likelihood: **confidence** measure.

Results

- Collected a **database.**
- Evaluation protocol: **Quantitative** results.

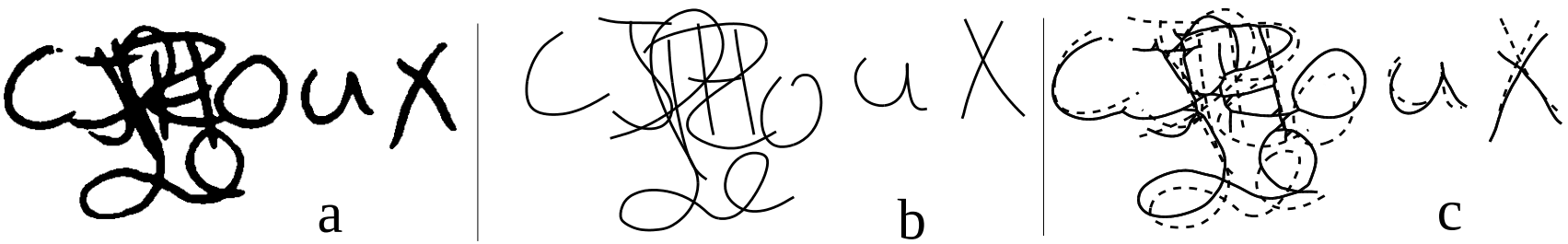
A static script to unravel

(a) Scanned binary signature.

(b) Typical dynamic exemplar of (a).

(c) Thinned version of (a) (solid) superimposed on (b) (dashed).

- Note the ambiguous regions in (a).

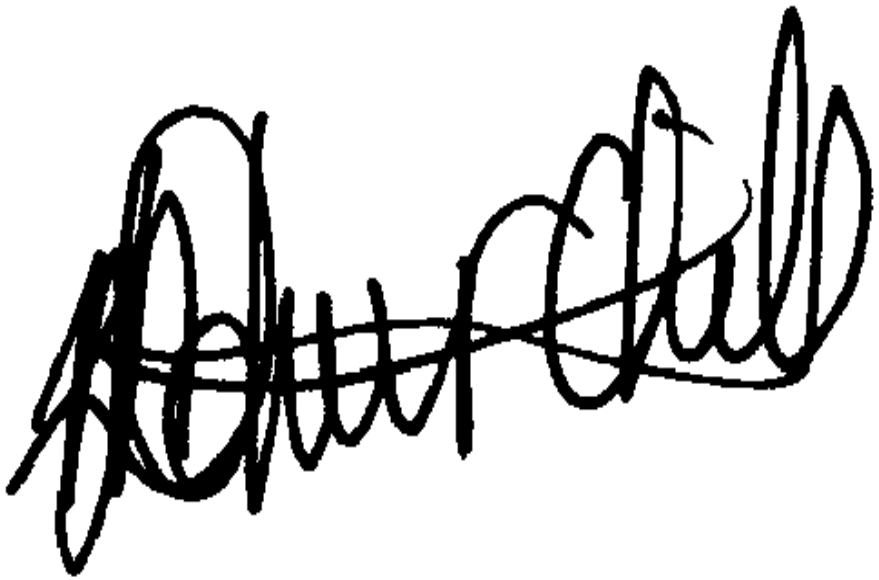


Thinning

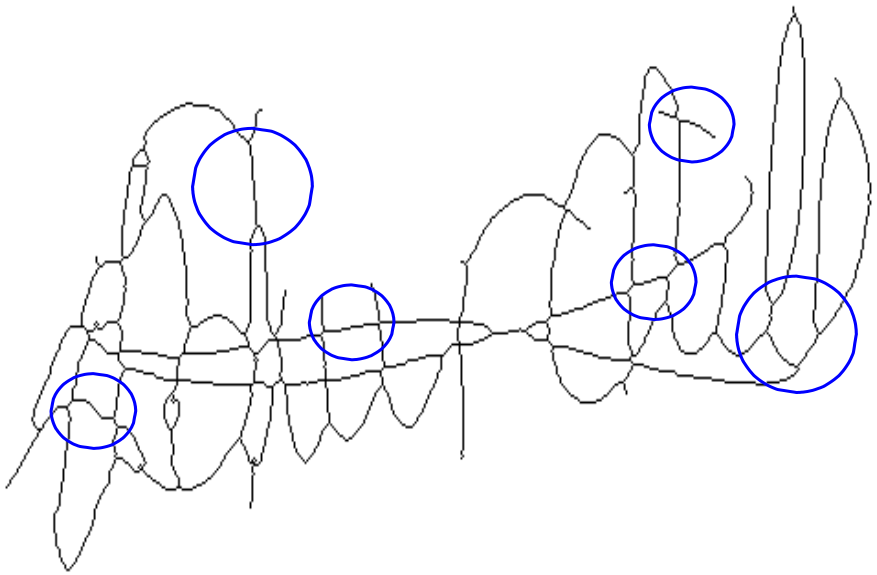
- Skeleton: Coincides with centreline of image:
  - Script → parametric curve.
  - Connectivity → HMM topology.

Standard thinning

- Compute centreline: artifacts are inevitable.



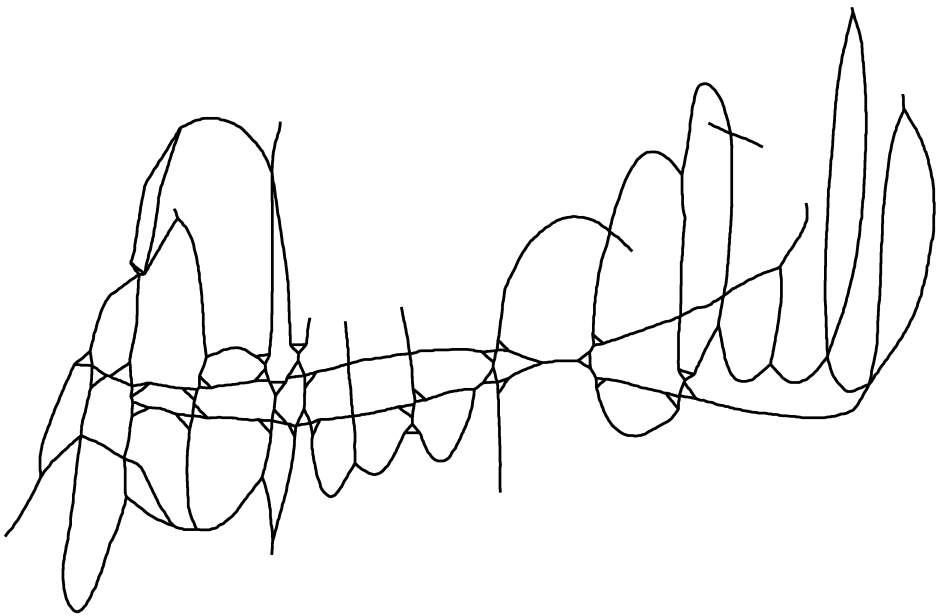
(a)

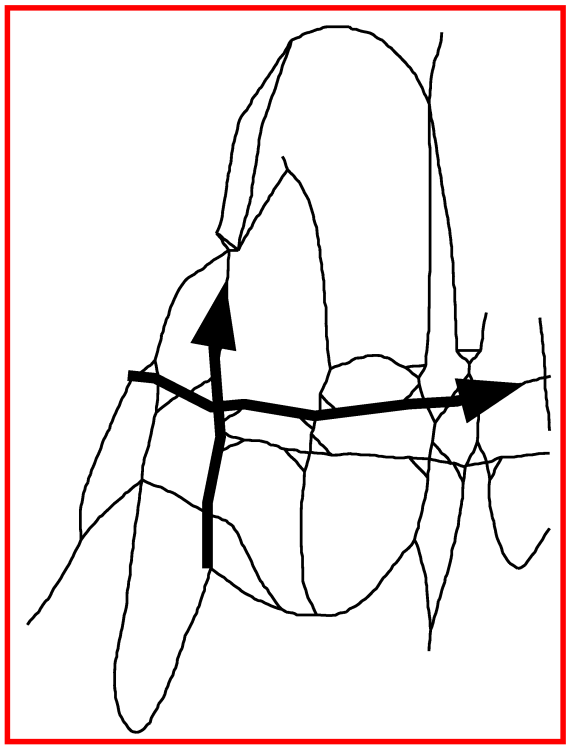


(b)

Application-specific skeleton

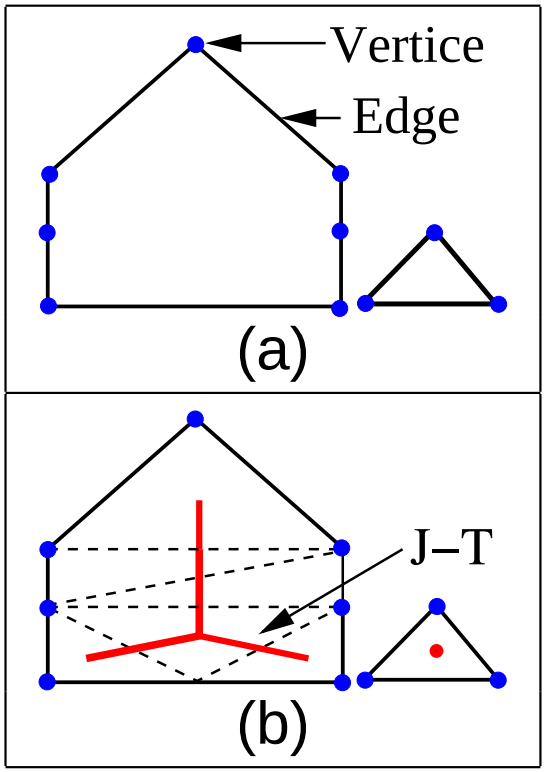
- Remove artifacts and smooth lines.
- Identify complicated regions.





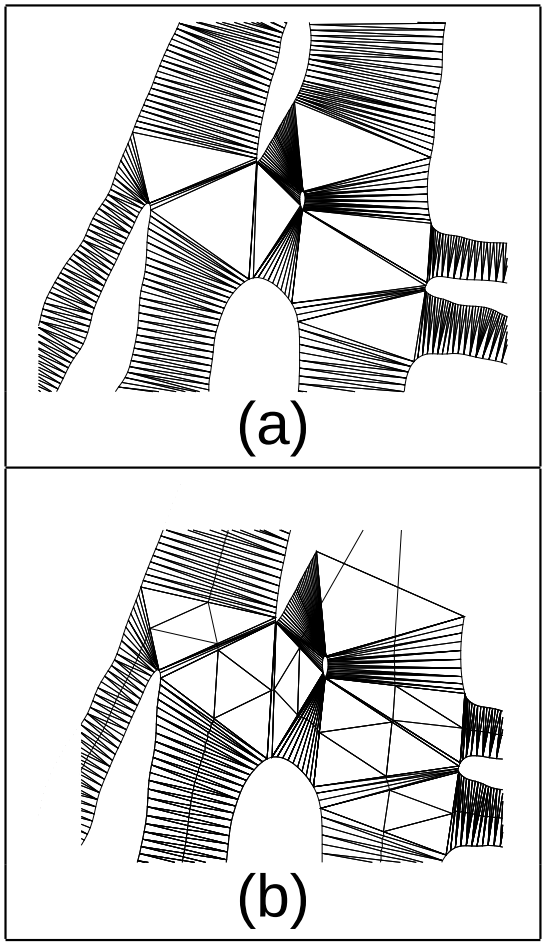
Shape partitioning

- Compute boundary.
- Boundary → approximating polygon.
- Partition polygon into subshapes.
- Employ Delaunay triangulation.
- Classify triangles.
- Triangles → standard skeleton.
- *Junction triangle (J-T)* contains intersection point.



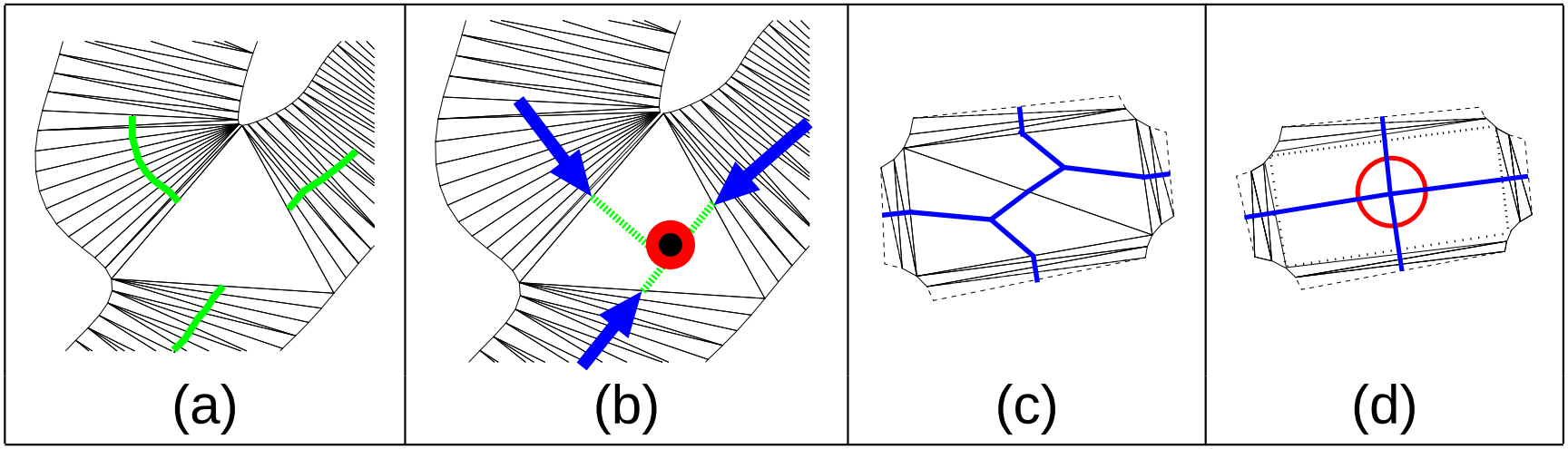
Identify complicated parts

- Many J-Ts within close proximity.
- Multiple crossings: no crucial choices.
- Introduce web-like structures.
- Smooth transitions slightly.
- Many lines: unravel using more powerful statistical methods.



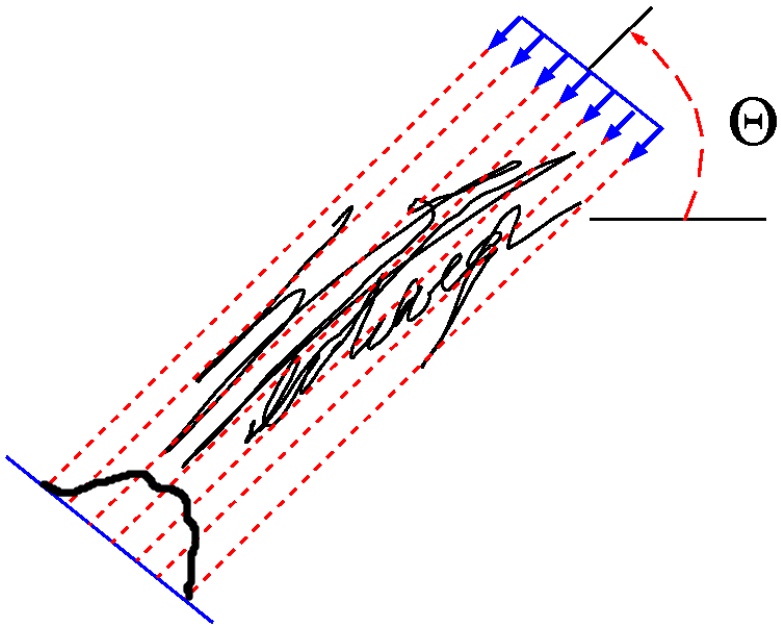
Remove artifacts in uncomplicated parts

- Splines estimate line directions entering J-Ts.
- Recalculate intersection points (inside J-Ts.)



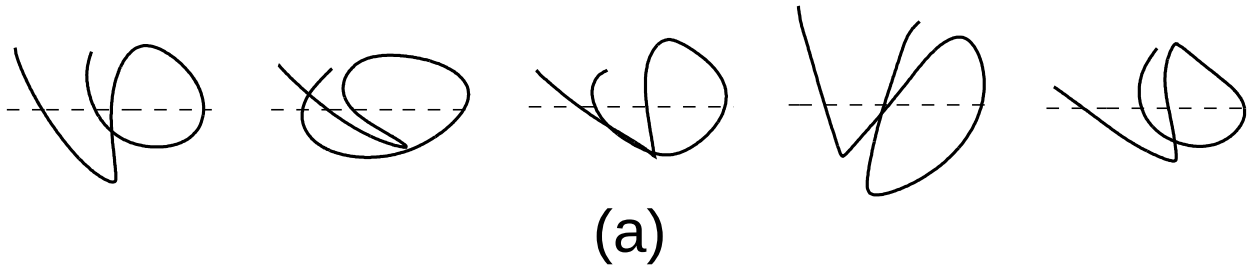
Orientation normalisation: Radon transform

- Align general orientation of skeleton and exemplar.
- Rotate (x, y) plane with θ degrees.
- **Projection:** All integrals along lines parallel to rotated x-axis.
- **Radon transform :** All projections for $\theta \in [0°, 180°]$.



Orientation normalisation: Radon transform

- Compute Radon transform of exemplar and skeleton.
- Minimise Euclidean distance.

PCA	 (a)
Radon	 (b)

HMM background

- N emitting states:
  - Associated Probability Density Functions (PDFs).
  - PDF observation likelihoods reflect similarities.
- Topology:
  - Defines interconnections between states.
  - Manipulated through transition link probabilities.
  - Governs the choices of pen trajectories.

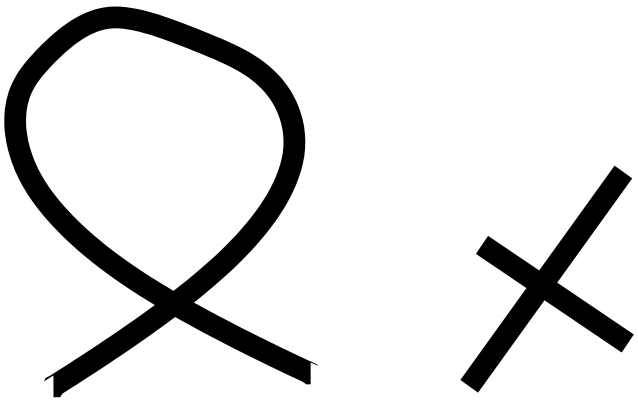
HMM background

- Order of an HMM:
  - Specifies state history.
  - Second-order HMMs model pen direction.
- Hidden state sequence:
  - Revealed after match.
  - State sequence → pen trajectory.

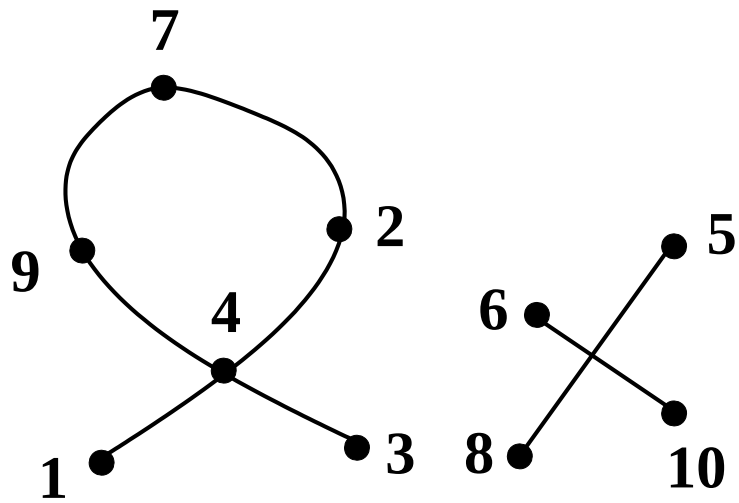
First-order HMM: Introductory example

(a) Static signature.

(b) Random-order skeleton samples.



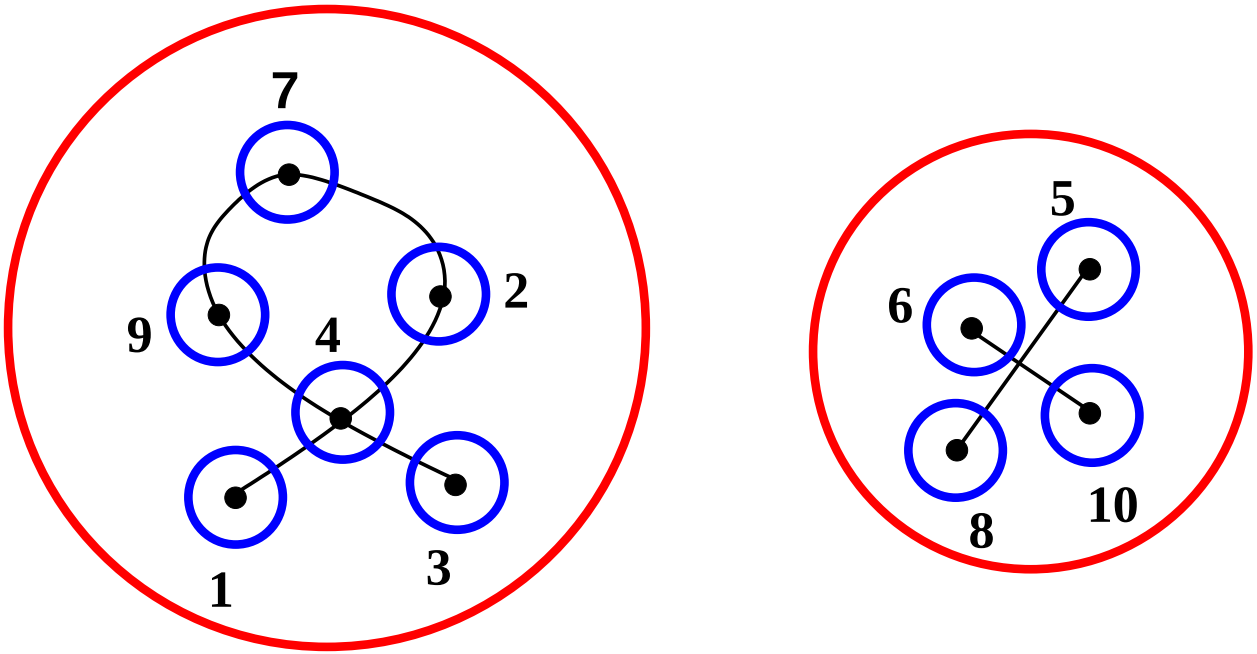
(a)



(b)

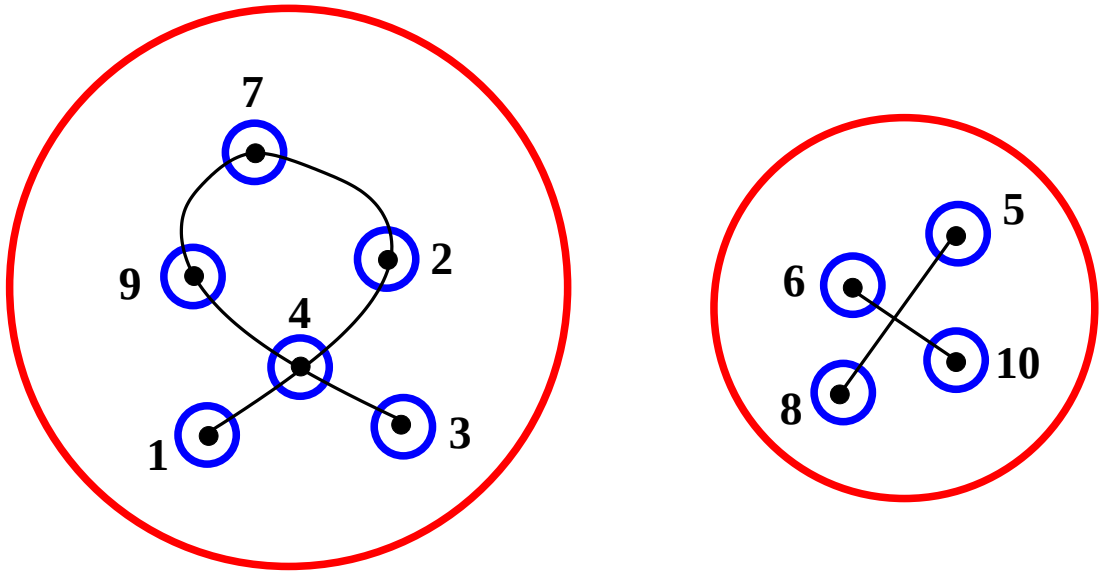
Identify sub-images

- Identify sub-images (disconnected parts).
- First-order HMM for each sub-image (red circle).
- **Note:** Simplified intersection.



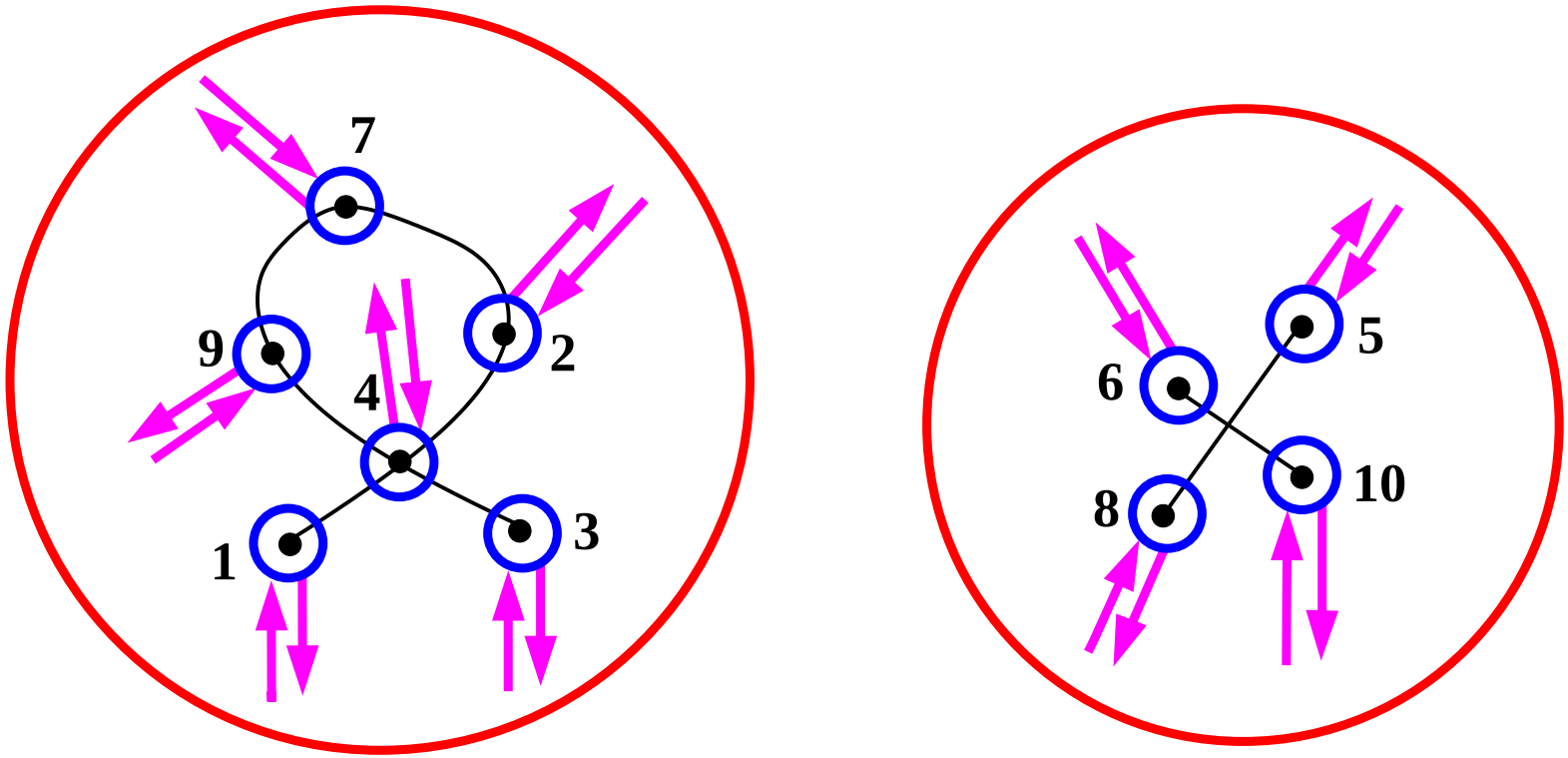
First-order HMM states

- Emitting state i (blue circle) for each skeleton sample $\mathbf{p}_i$.
- Each emitting state has an associated observation PDF.
- PDF is circular Gaussian $\mathcal{N}(\boldsymbol{\mu}, \sigma)$.
- $\mathcal{N}(\boldsymbol{\mu}_i, \sigma_i)$ "remembers" skeleton sample of state i: $\boldsymbol{\mu}_i = \mathbf{p}_i$.



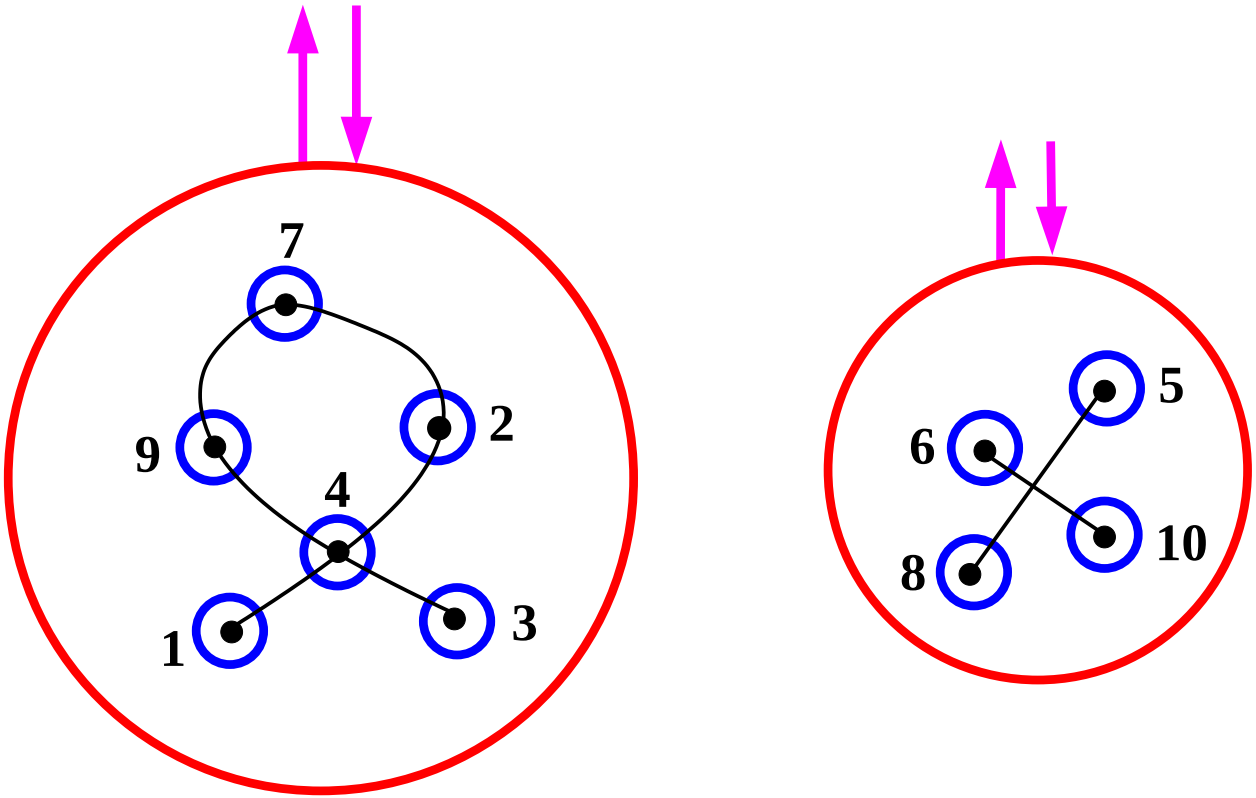
Starting and terminating positions

- Enter and exit the HMM at any state (pink arrows.)
- Trajectory can start/terminate at any skeleton sample.



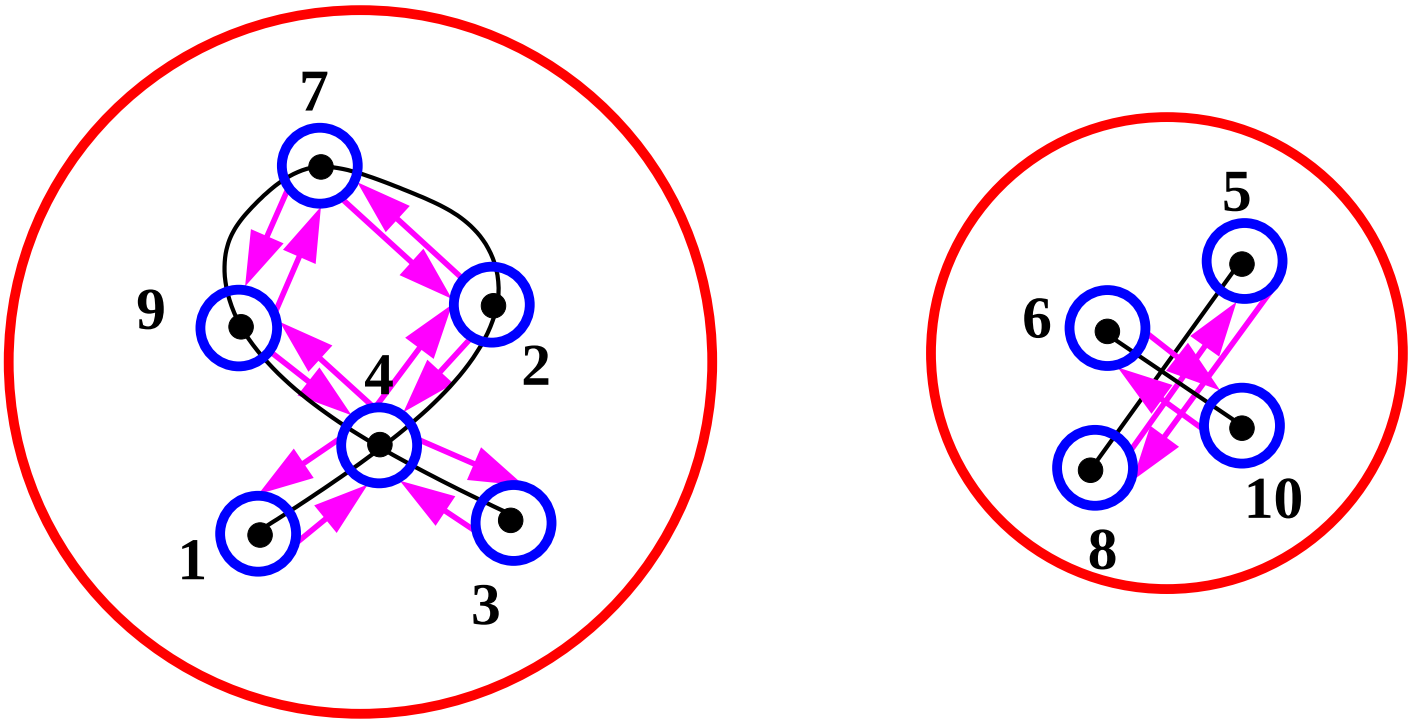
Starting and terminating positions

- **Simplify:** replace links with two big arrows.



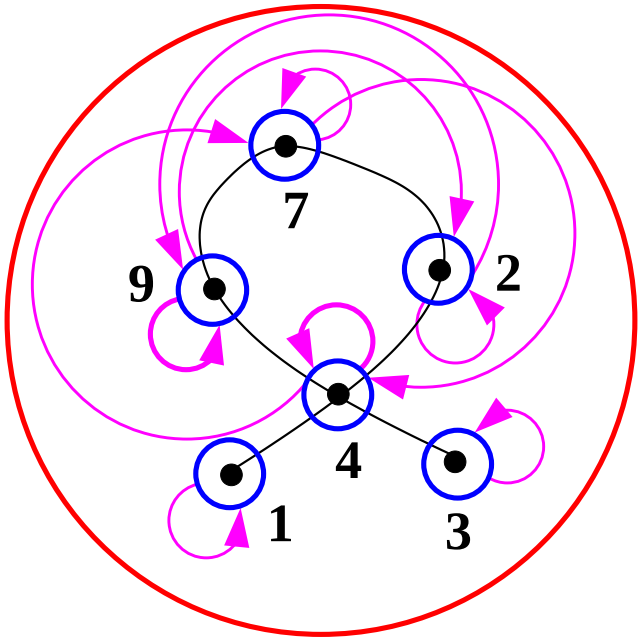
Connect neighbouring states

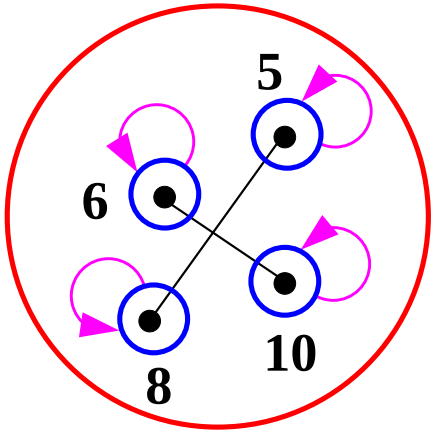
- Connect **neighbouring** states.
- Neighbouring states have **adjacent** skeleton samples.



Compensate for signature length variations

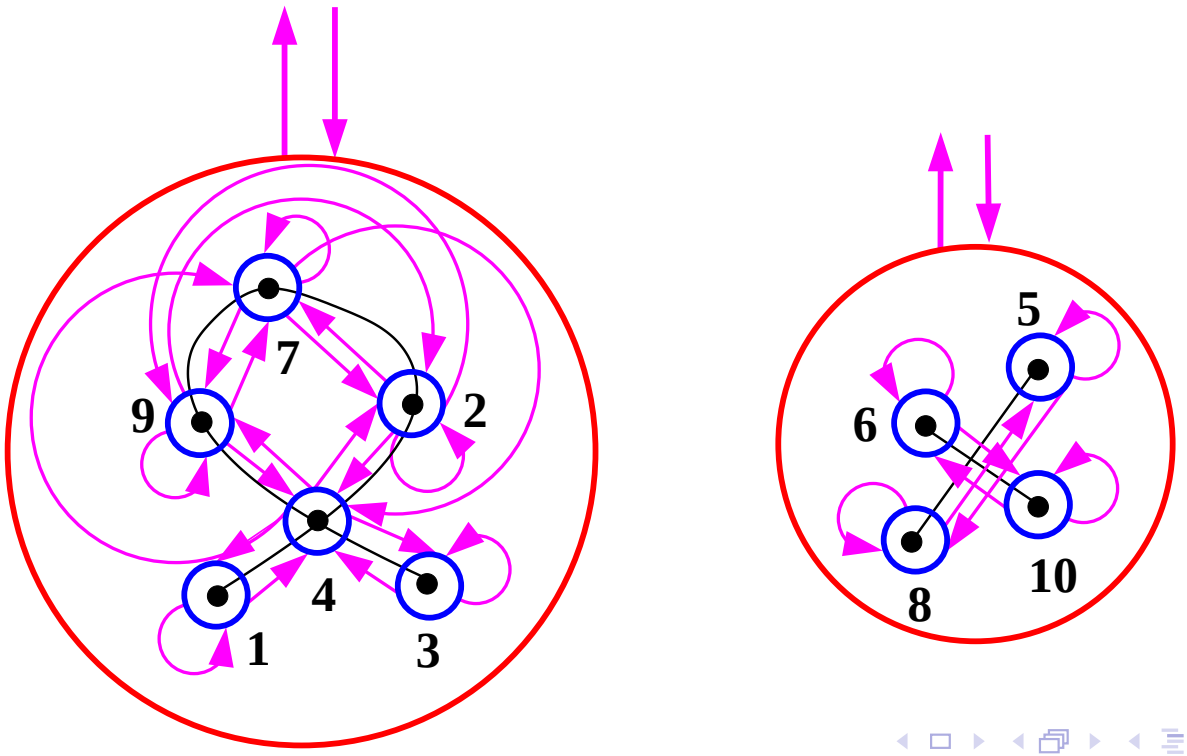
- Compensate for different numbers of samples.
- **Selfloops:** Connect a state to itself.
- **Skiplinks:** Connect neighbours of states with two neighbours.





First-order HMM results

- Results: unsatisfactory, due to **loss of context.**
- **Local pen directions** unknown.
- Any point can be a turning point.



Deriving a second-order HMM

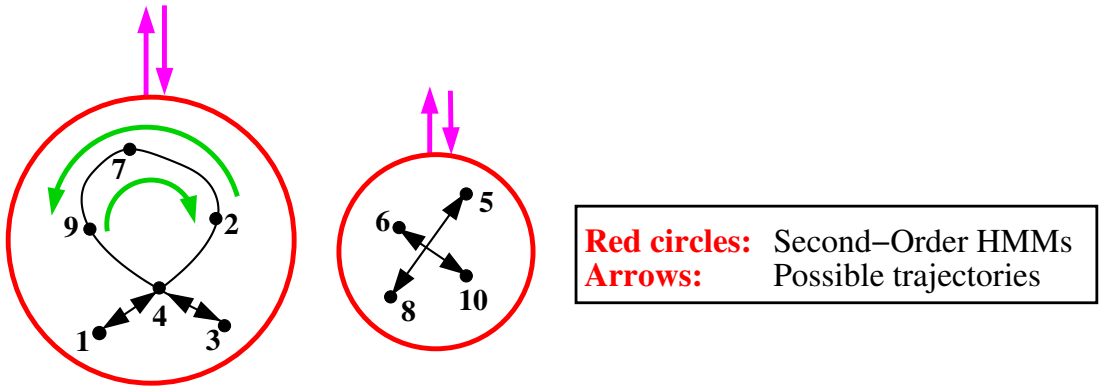
- **First-order HMM:** $\boldsymbol{\mu}_i = \mathbf{p}_i$.
- Unknown previous skeleton sample at state i—no direction.
- Any point can be a turning point.
- **Second-order HMM:** Includes more context.
- "Remember" the previous state.
- Any state is preceded only by states that share the same PDF.

Deriving a second-order HMM

- Crucial advantage: each state is preceded only by states that share the same skeleton sample.
- State i is redefined:
  1. $\boldsymbol{\mu}_i = \mathbf{p}_i$.
  2. $\mathbf{p}_{ki}$ (unique previous skeleton sample k.)
- Include unambiguous directional/normalised velocity component at each state $\frac{\boldsymbol{\mu}_i - \mathbf{p}_{ki}}{\|\boldsymbol{\mu}_i - \mathbf{p}_{ki}\|}$.
- Manipulate transition links: model turning points.

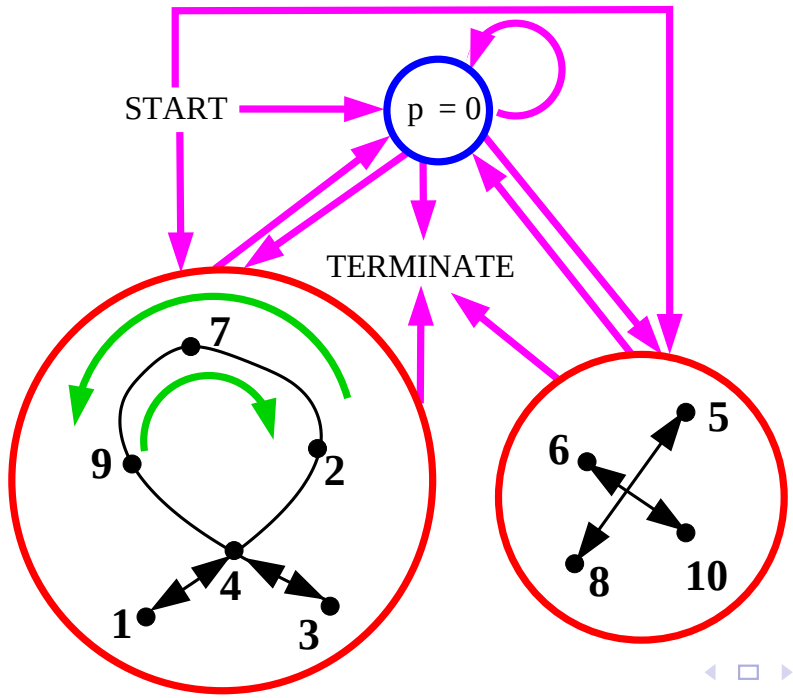
Example: Second-order HMM

- **Red circles:** second-order HMMs.
- **Arrows:** pen trajectories.
- Start/terminate at any skeleton sample.
- Maintain direction of traversal *between* intersection/turning points.



Connecting the sub-image HMMs

- Construct hierarchical HMM (**HHMM**).
- **State** for each second-order HMM.
- Additional ***zero pressure state.***
- Model pen-up/pen-down events.

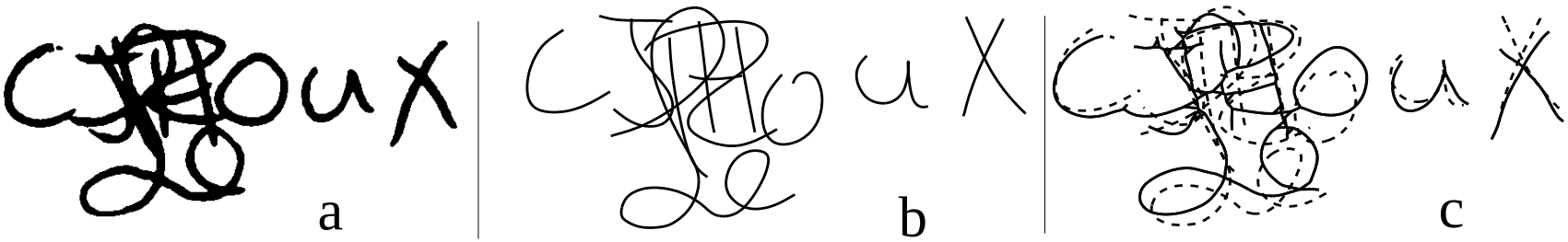


Estimating the pen trajectory

1. Match all dynamic exemplars of individual to HHMM.
2. Viterbi algorithm searches for most likely state sequence.
3. Dynamic exemplar sample $\rightarrow$ state.
4. Each state $\rightarrow$ skeleton sample.
5. Sequence of states $\rightarrow$ desired sequence of skeleton samples (pen trajectory).
6. Likelihood δ: confidence measure.
   - Indication of trajectory accuracy.
   - Identify forgeries in signature verification application.

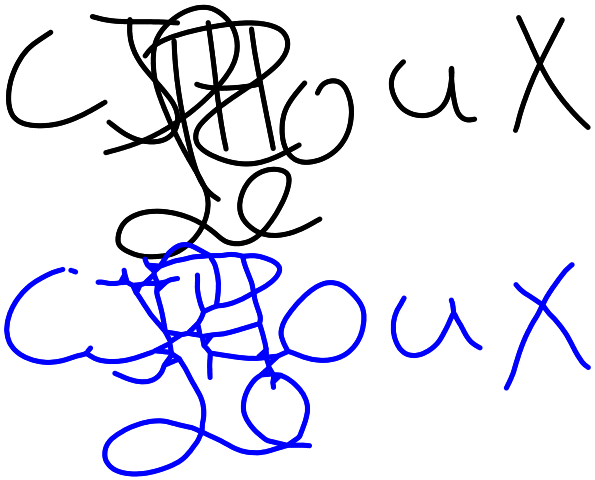
Example: matched signatures

- The dynamic exemplar (b) has the highest likelihood.
- Match (b) to the HHMM of the skeleton (solid line) in (c).



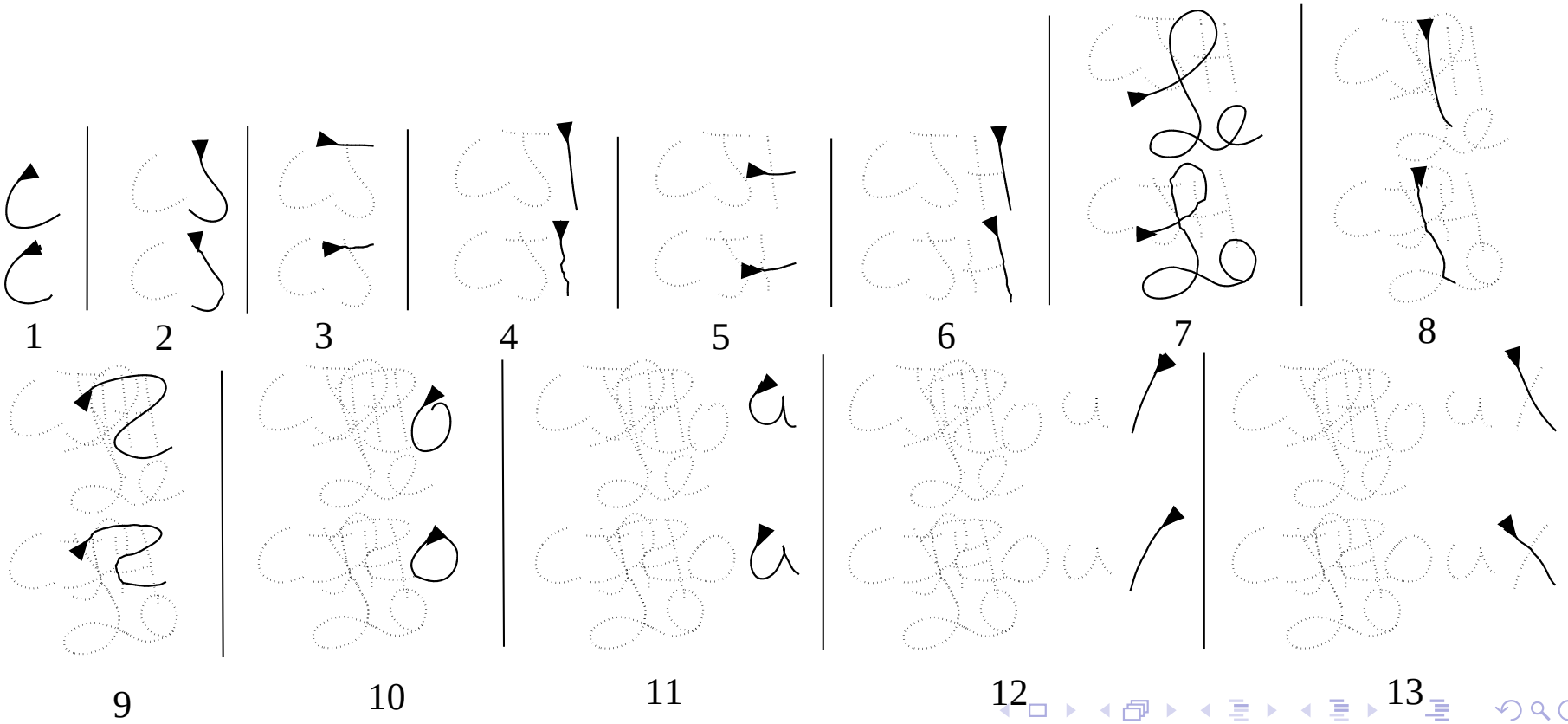
Animation

- Dynamic exemplar (top).
  - Pen trajectory is known.
  - Different signature from static image.
- Static skeleton (bottom).
  - Pen trajectory is unknown.
- Animated circles show pointwise correspondence.



Examples: animation figure

- 13 trajectories extracted from 3 sub-images.
- 3 – 9 extracted from ambiguous middle region.



Evaluating typical errors

- Dissimilarities cause errors.
- Collect database.
- Record static script and dynamic counterpart simultaneously.
- Dynamic counterpart → ground truth.
- **Note:** ground truth only for evaluation purposes.

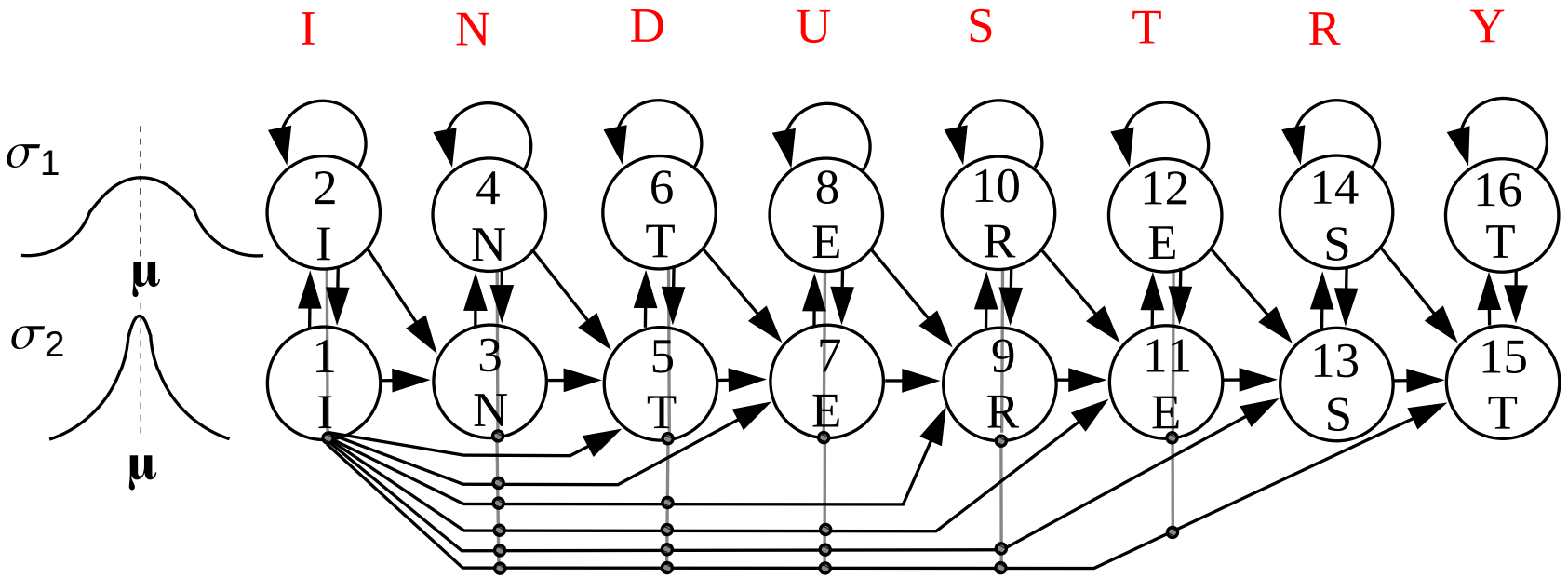
Evaluating typical errors

Evaluating efficacy:

- Compare estimated trajectory and ground truth.
- Sequences have different lengths.
- Pointwise comparison is not possible.

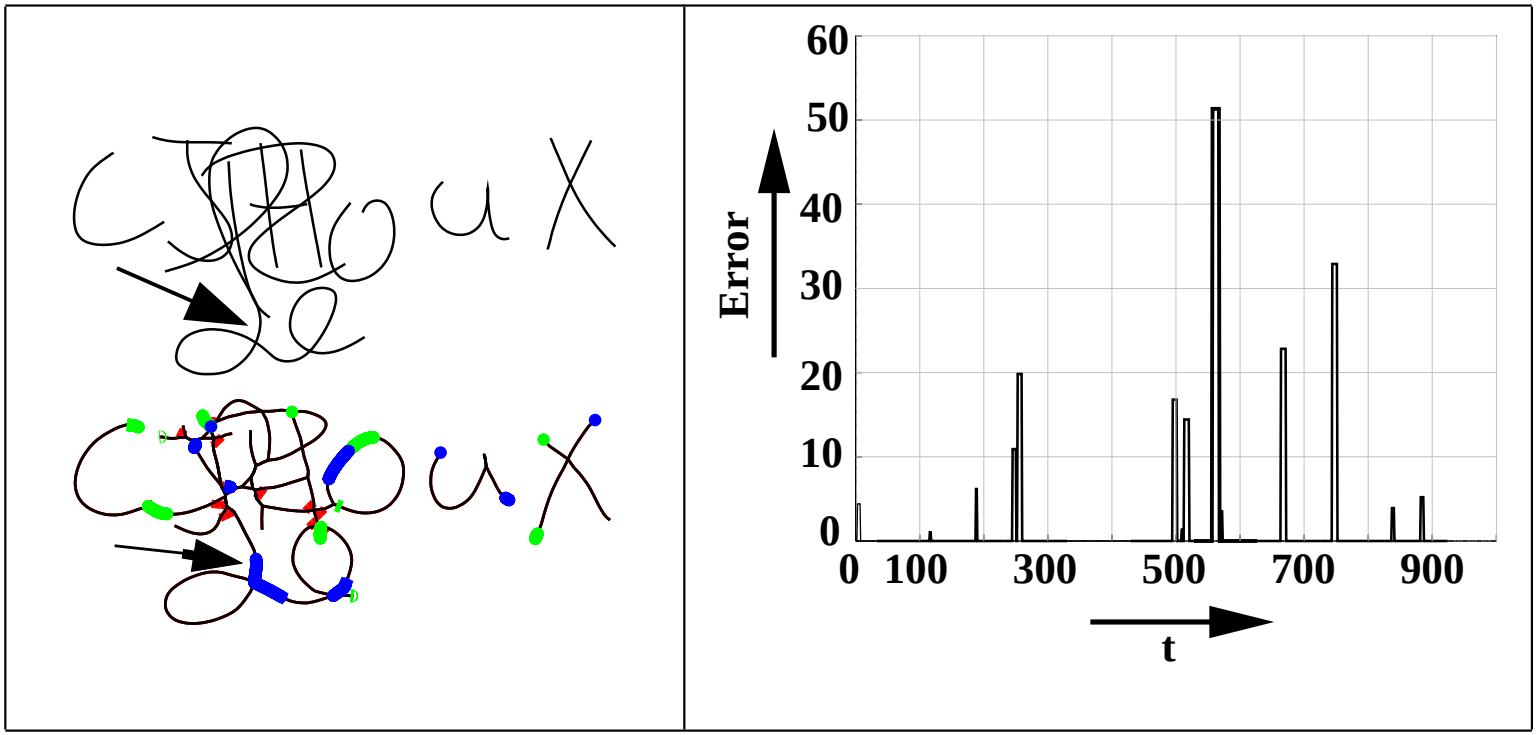
Quantitative measurements

Establish pointwise correspondence: hypothetical example.



- $\mathbf{s} = [1, 3, 4, 4, 13, 15, 16, 16]$ or $\mathbf{s} = [1, 3, 4, 4, 4, 5, 9, 10]$.
- Depends on 2D coordinates.
- Errors: Expressed as percentage of ground-truth path length.

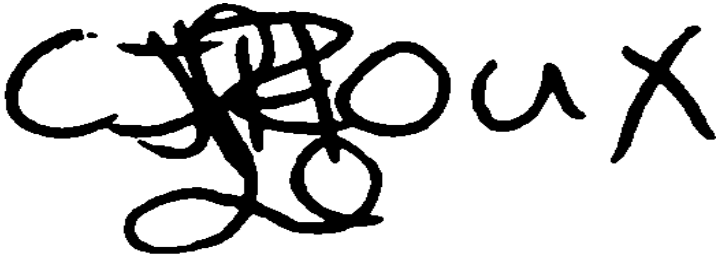
Experimental results and examples



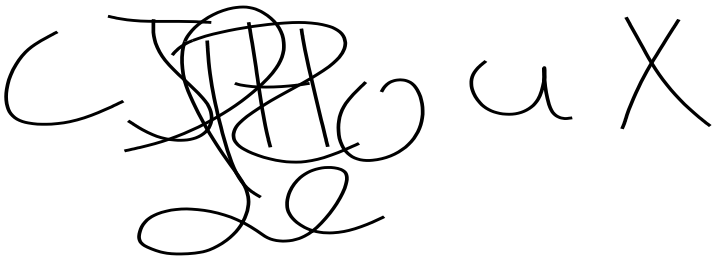
- **Pulse heights** quantify erroneous curves.
- $a_{\text{traj}} = 93\%$
- $\log(\delta) = -8.15$
- $a_{\text{total}} = 88\%$ averaged over 51 users.

Results and examples

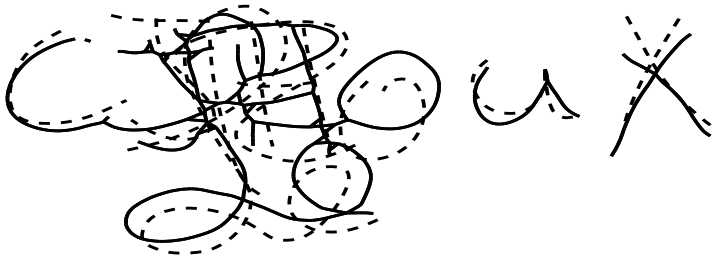
Static signature



Dynamic exemplar



Dynamic Exemplar (––) and static skeleton (–)



Red Dots : **Corresponding points**
Yellow Dots : **Error inception points**

View ground–truth pen trajectory
Top : **Dynamic Counterpart**
Bottom : **Static skeleton**

View estimated pen trajectory
Top : **Dynamic Exemplar**
Bottom : **Static skeleton**

View evaluation trajectories
Top : **Error function**
Bottom : **Ground–truth and estimated trajectories**

Practical issue: No corresponding curves.

Jump to Conclusion

Absent dynamic curves.

- Curve in static → no curve in dynamic.
- Each dynamic exemplar sample → skeleton sample.
- High likelihood δ falsely reflects high accuracy.
- **Example:** dynamic “1” → static “7”.
- General handwriting variations.
- Insufficient pressure levels: e.g., 256 vs. 1024 levels.

Practical issue: No corresponding curves. 51

Absent static curves (broken lines).

- Image discontinuities at non-zero pressure values.
- Inaccurate trajectories.
- Dry ink / binarisation of low intensities.

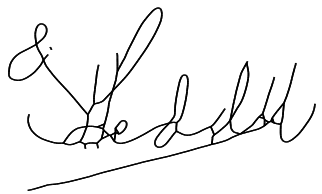
Solution to absent dynamic curves.

- Likelihood δ for each dynamic exemplar state sequence.
- Weighted δ:
  - $\log(\delta_{\mathrm{W}}) = \mathrm{sign}(\log(\delta))\frac{R_{\mathrm{L}}}{T_{\mathrm{L}}} \mid \log(\delta) \mid$
  - T_{L} = total static skeleton path length.
  - R_{L} = extracted skeleton path length.
- $\delta \rightarrow$ insertions and substitutions.
- $\frac{R_{\mathrm{L}}}{T_{\mathrm{L}}} \rightarrow$ deletions.
- $\delta_{\mathrm{W}} \rightarrow$ trajectory accuracy a_{c}.

Example: absent dynamic curves.

- δ_W increases class separability.

(a)



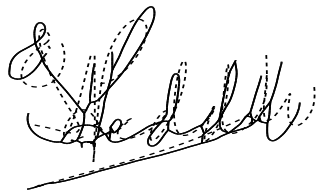
(b)

$\log(\delta) = -8.75$
$\frac{R_L}{T_L} = 0.9511$
$\log(\delta_W) = -9.2$
$a_C = 83.8\%$

(c)

$\log(\delta) = -18.56$
$\frac{R_L}{T_L} = 0.474$
$\log(\delta_W) = -39.18$
$a_C = -0.7\%$

(d)



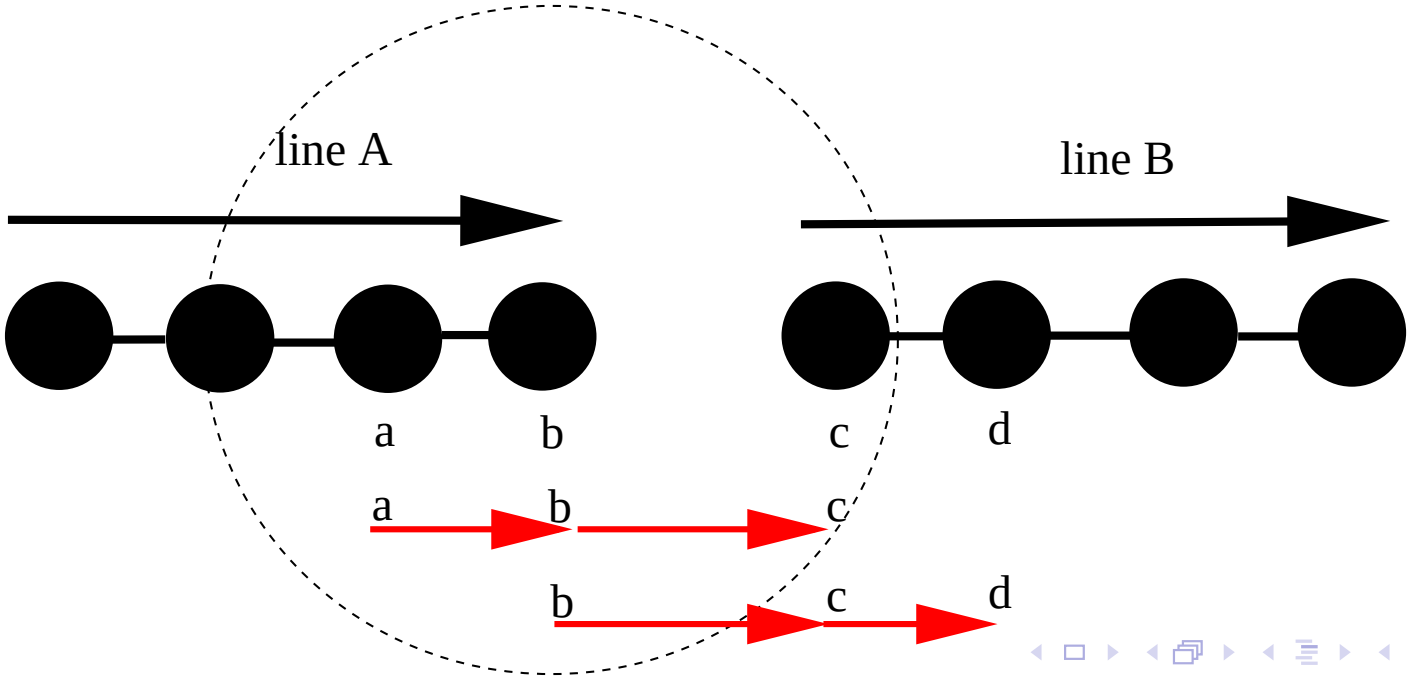
(e)

(f)

- Note: possibilities for off-line signature verification!

Broken lines in static image.

- Identify nearby endpoints (part of broken line.)
- Broken lines must have similar directions.
- Include additional states.
  - Indirectly connect endpoints.
  - Enable transitions between discontinuities.



Conclusion

Modelling static scripts statistically

- Estimate trajectories using **two-level HHMMs**.
- **Higher-level:** model **pen pressure**.
- **Lower-level:** high-order HMMs, non-zero pressure:
  - Model **continuous** pen motions.
  - Model **intersection/turning** points.
- HMMs robust to **variations**:
  - **PDFs:** model translation and rotational variations.
  - **Skiplinks, selfloops:** model length variations.

Conclusion

Estimating the pen trajectories

- **Viterbi algorithm:**
  - **Globally** optimised:
    - Resolve ambiguous **intersection/turning** points.
    - Identify **starting/terminating** positions.
  - **Efficient.**

Results

- Approximately 88% **accurate**.
- **Evaluation protocol**: **Quantitative** results.

Conclusion

Contributions:

Existing methods	Starting/ Terminating	Intersect/ Turning	Pressure	Efficient
Rule-based	√	×	√	√
Graph-theoretical	√	√	×	×
Local correspondence	√	×	√	√
Our method	√	√	√	√