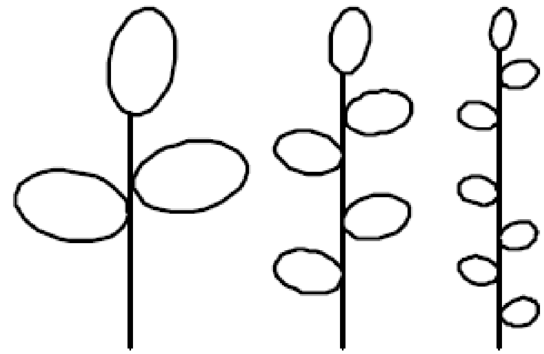
Detecting Shapes in Cluttered Images

CSE 6367 – Computer Vision
Vassilis Athitsos
University of Texas at Arlington

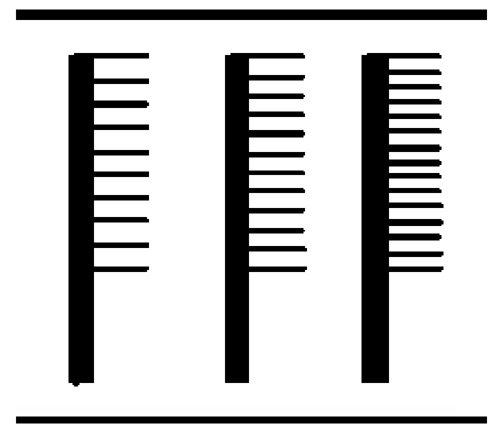
Objects With Variable Shape Structure

Objects with parts that can:

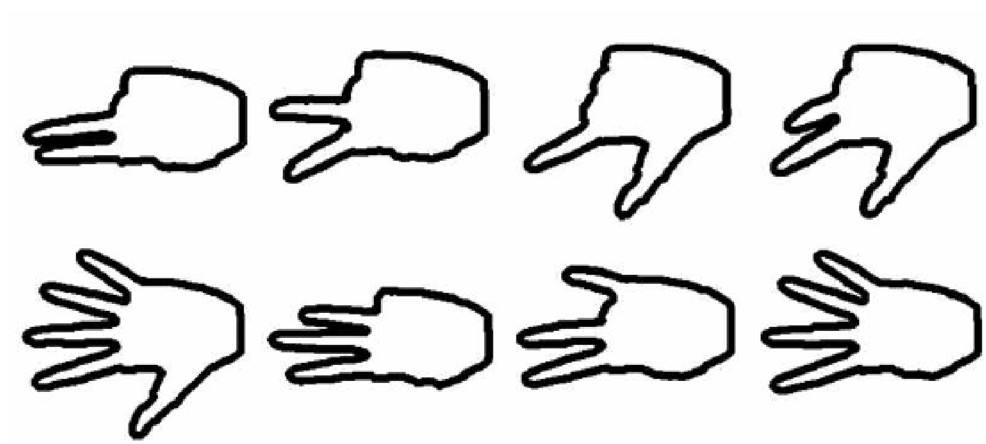
- Occur an unknown number of times.
  - Leaves, comb teeth.
- Be not present at all.
  - Leaves, comb teeth, fingers.
- Have alternative appearances.
  - Fingers.



branches



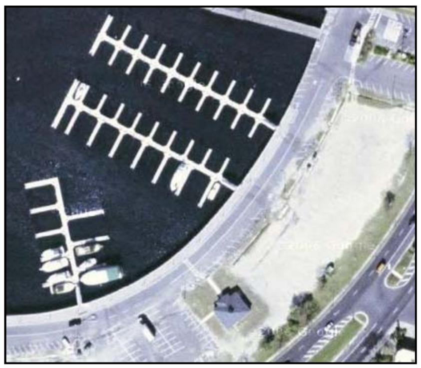
combs



hands

Other Examples

- Satellite Imaging

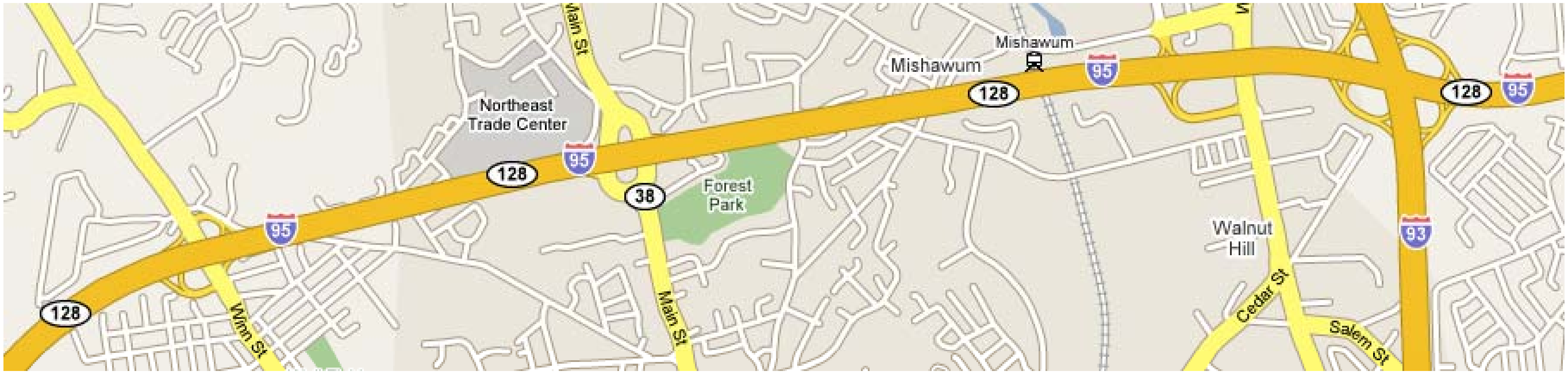


boat pier

parking lot

residential neighborhood

Roadways



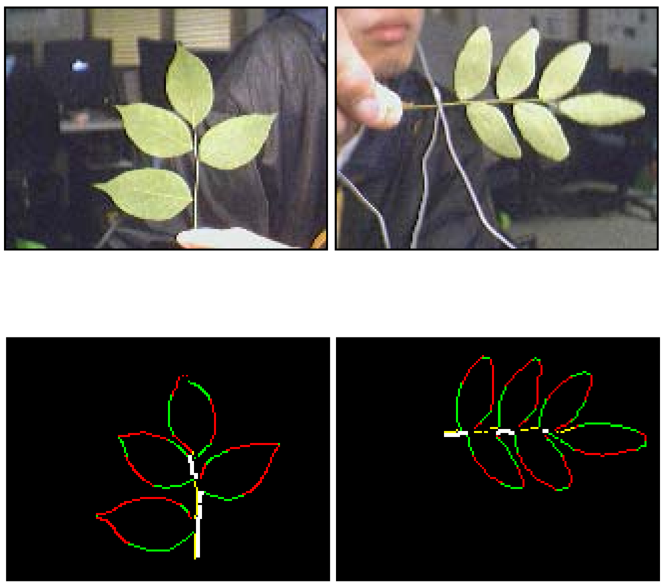
highway exits

Other Examples

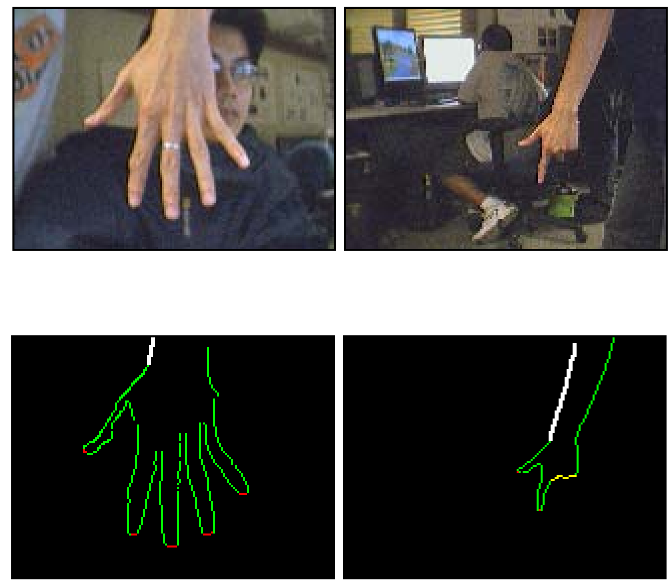
- Medical Imaging

ribs & bronchial tree

Goal: Detection in Cluttered Images



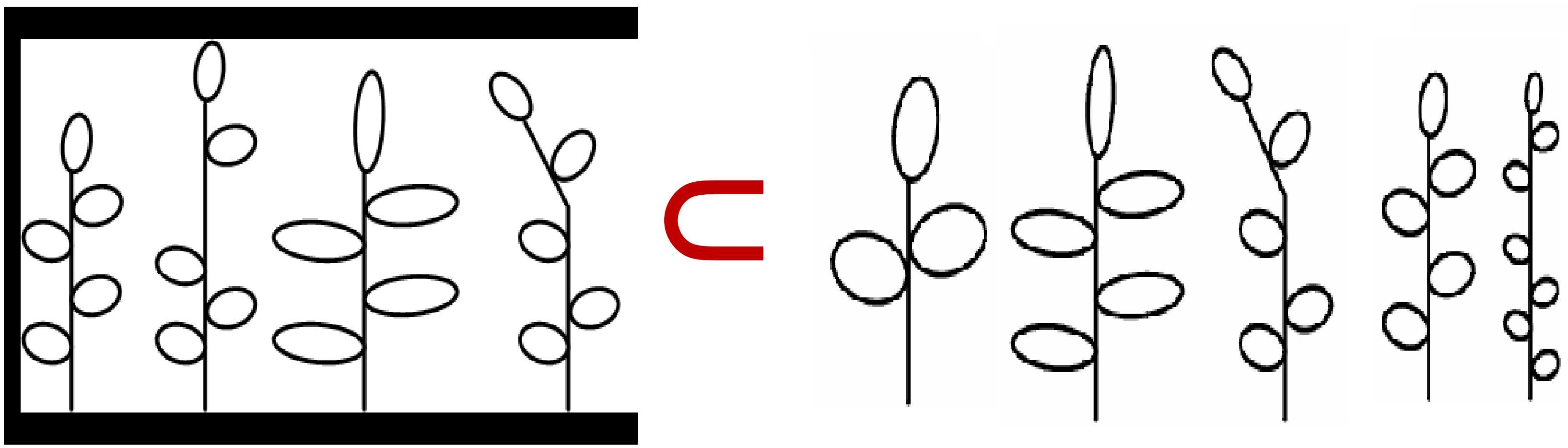
Branches



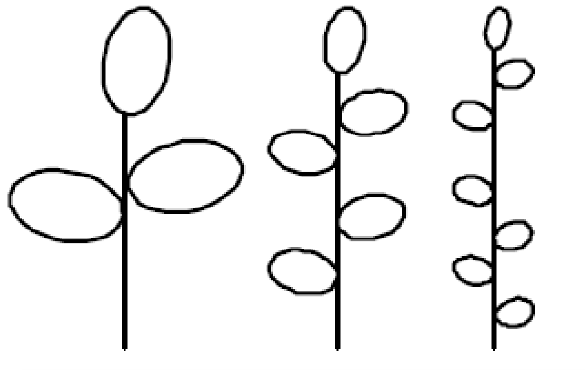
Hands

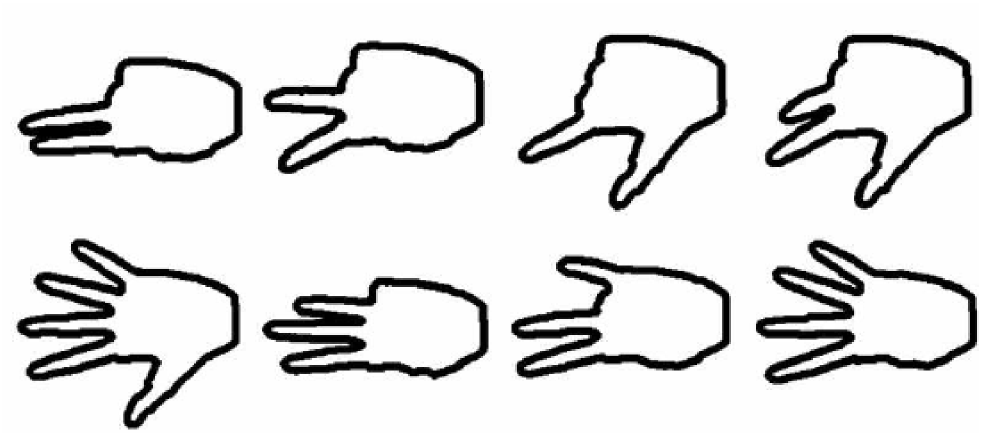
Deformable Models Not Adequate

- Shape Modeling:
  - "Shape deformation" ≠ "structure variation"



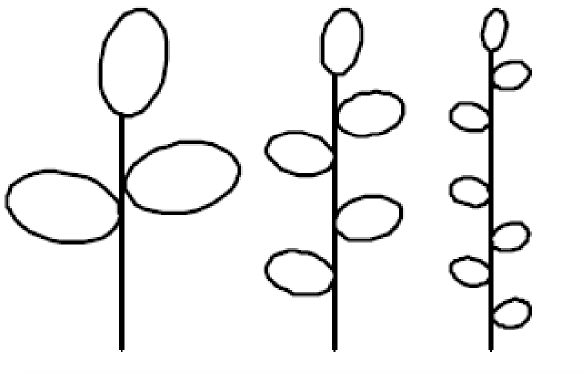
Limitations of Fixed-Structure Methods

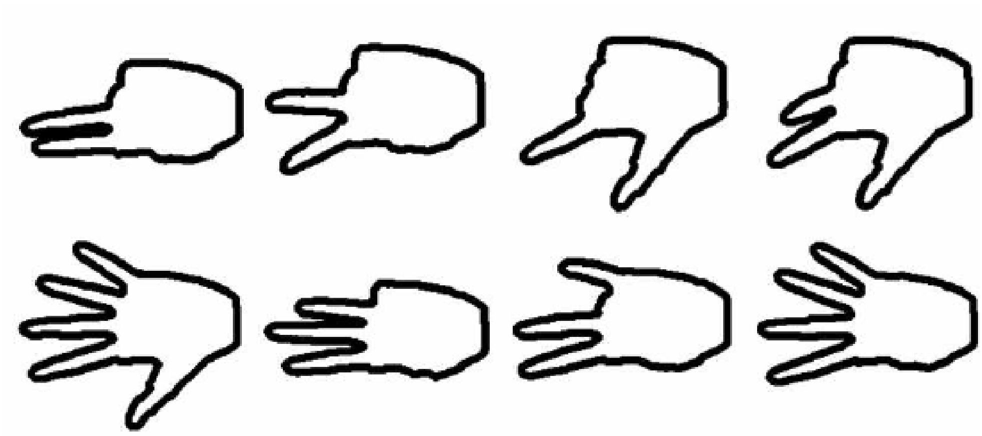




- A different model is needed for each fixed structure.

Limitations of Fixed-Structure Methods

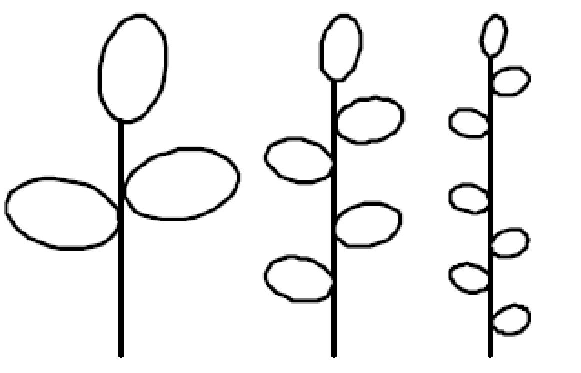


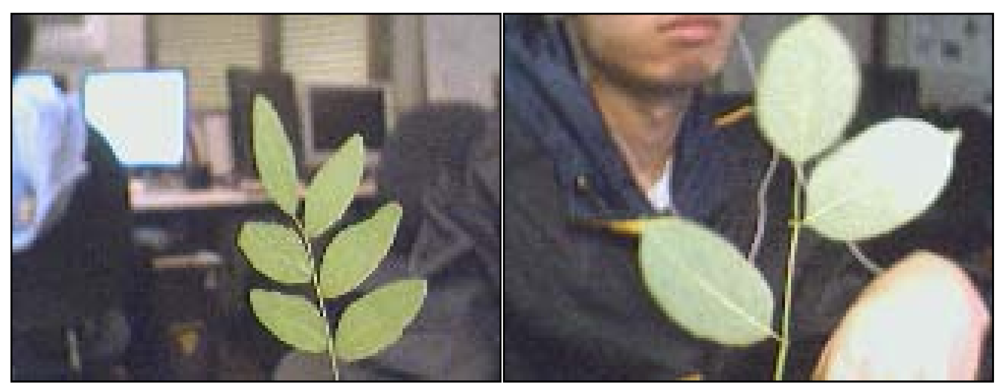


- Worst case: number of models is exponential to number of object parts.

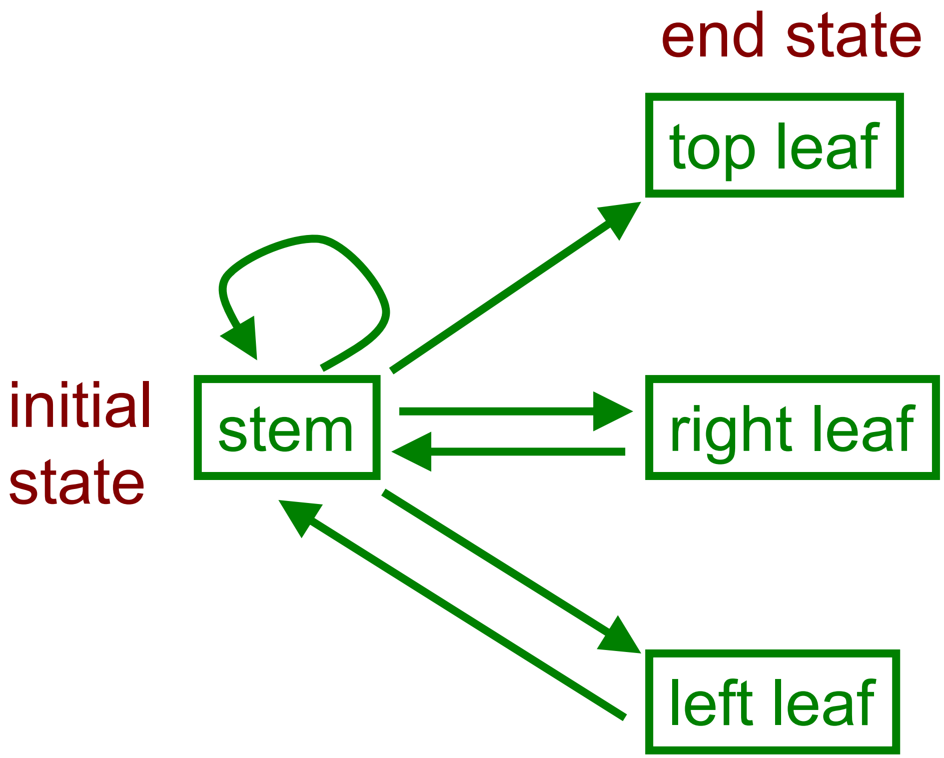
Overview of Approach

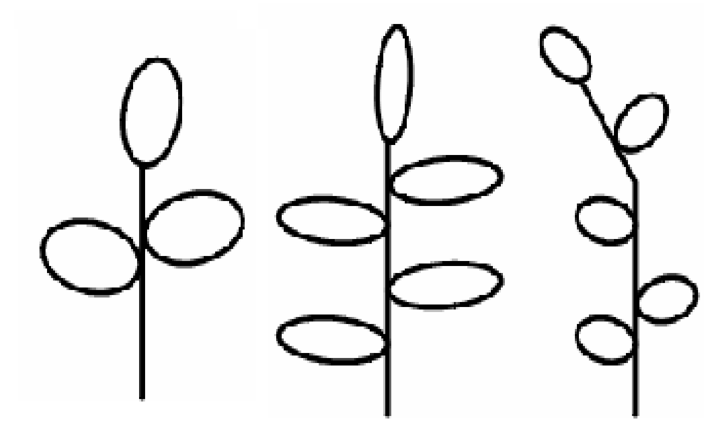
- Hidden Markov Models (HMMs) can *generate* shapes of variable structure.
- HMMs cannot be *matched* to an image.
  - Image features are not ordered.
  - Many (possibly most) features are clutter.
- Solution: Hidden State Shape Models.





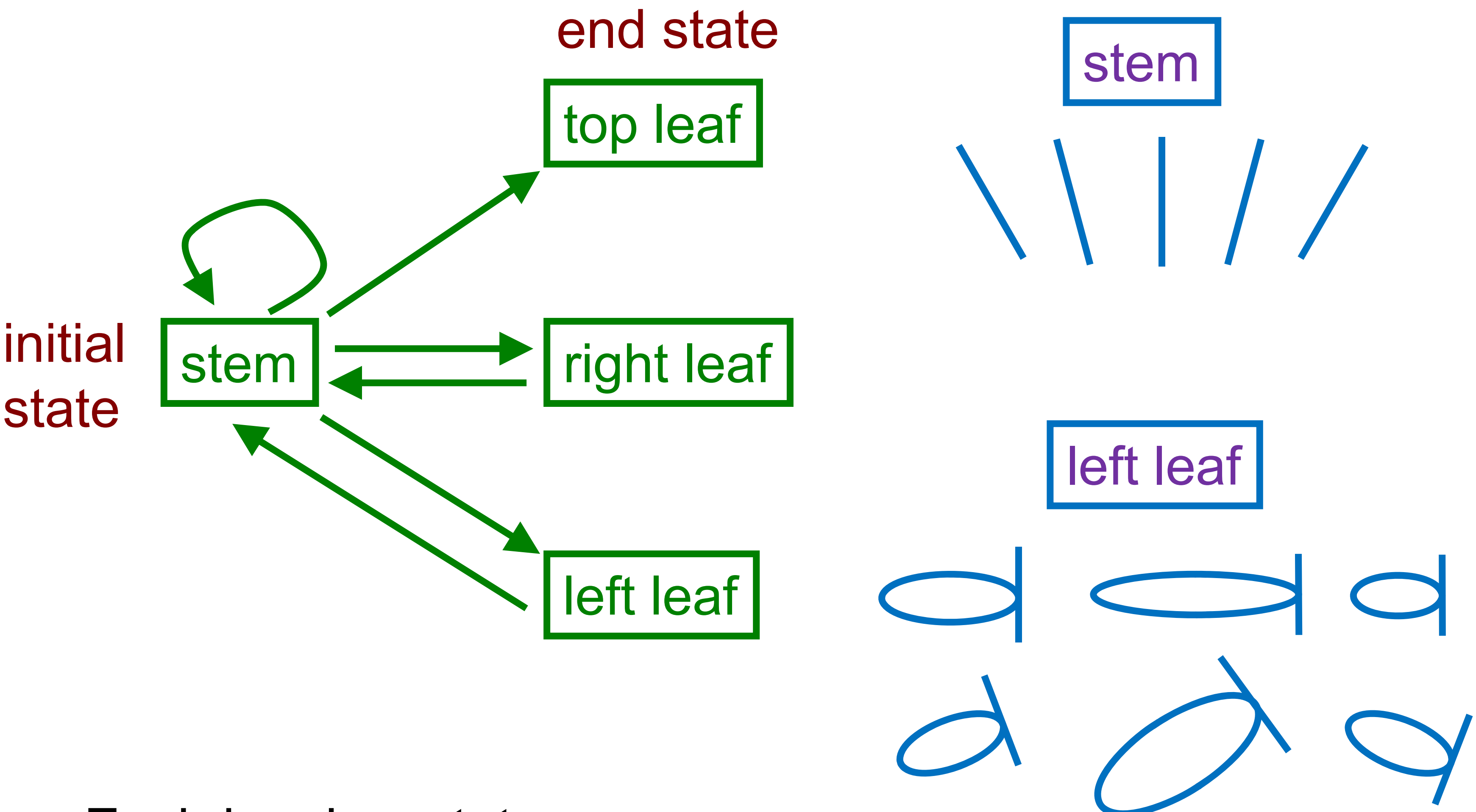
An HMM for Branches of Leaves





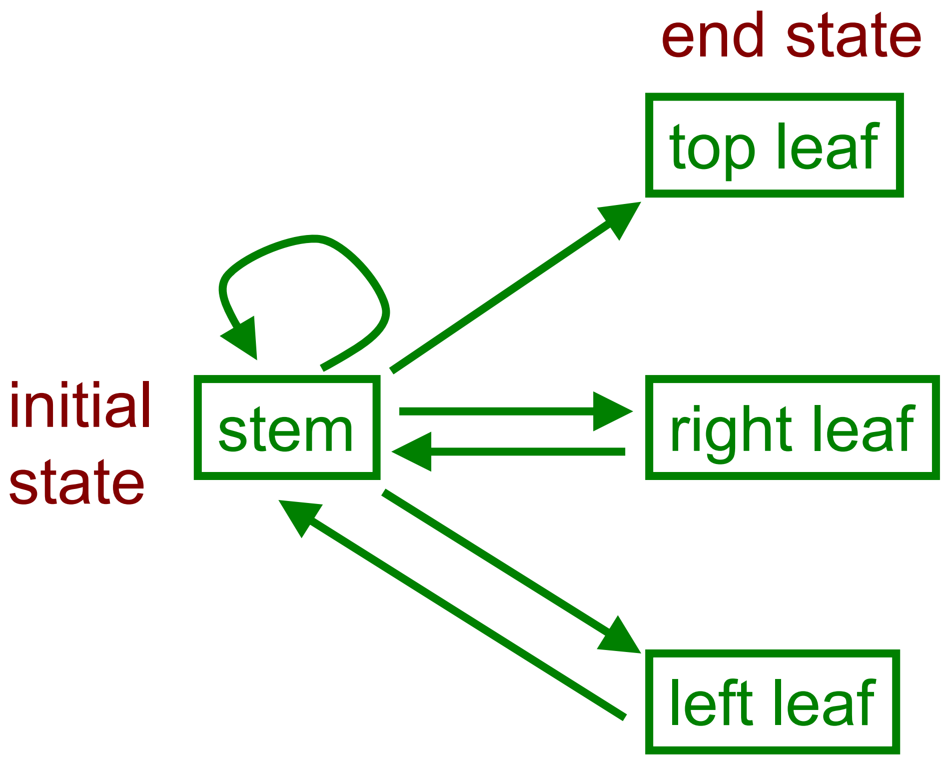
- Each box is a state.
- Arrows show legal state transitions.

An HMM for Branches of Leaves



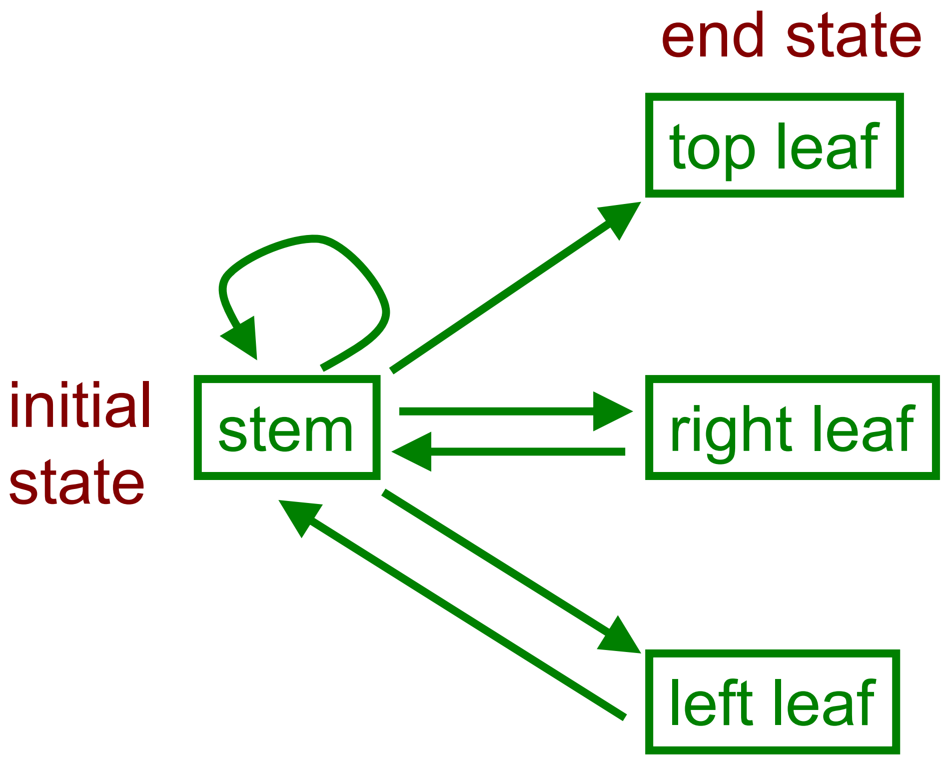
- Each box is a state.
- For each state there is a probability distribution of shape appearance.

Generating a Branch



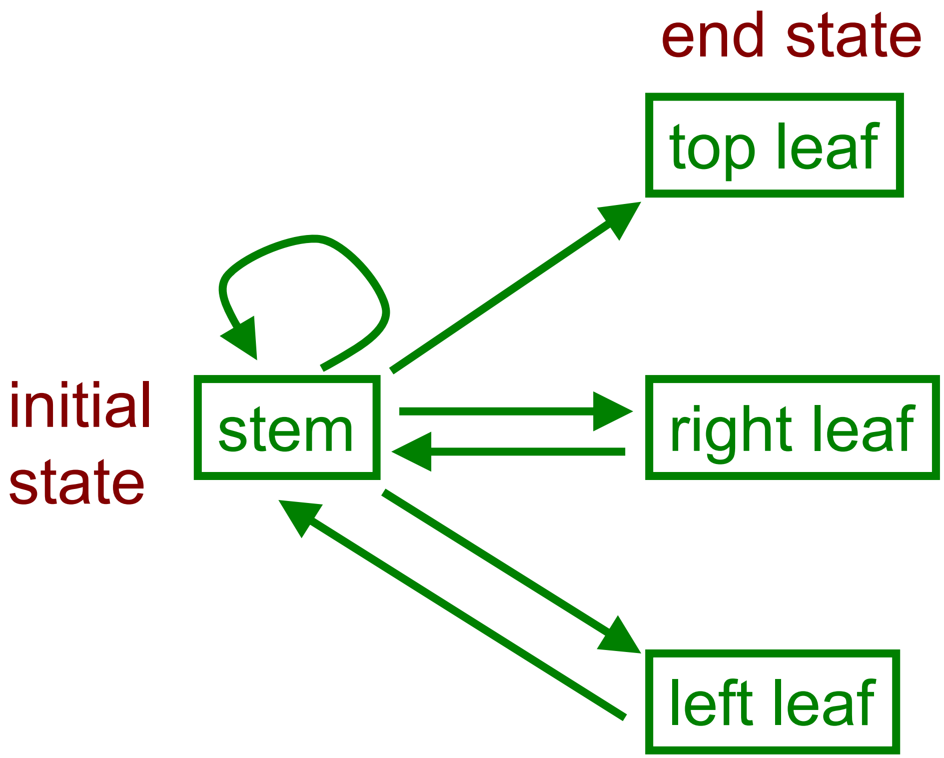
- Choose a legal initial state.
- Choose a shape from the state's shape distribution.

Generating a Branch

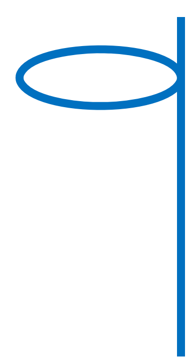


- Choose a legal initial state.
- Choose a shape from the state's shape distribution.

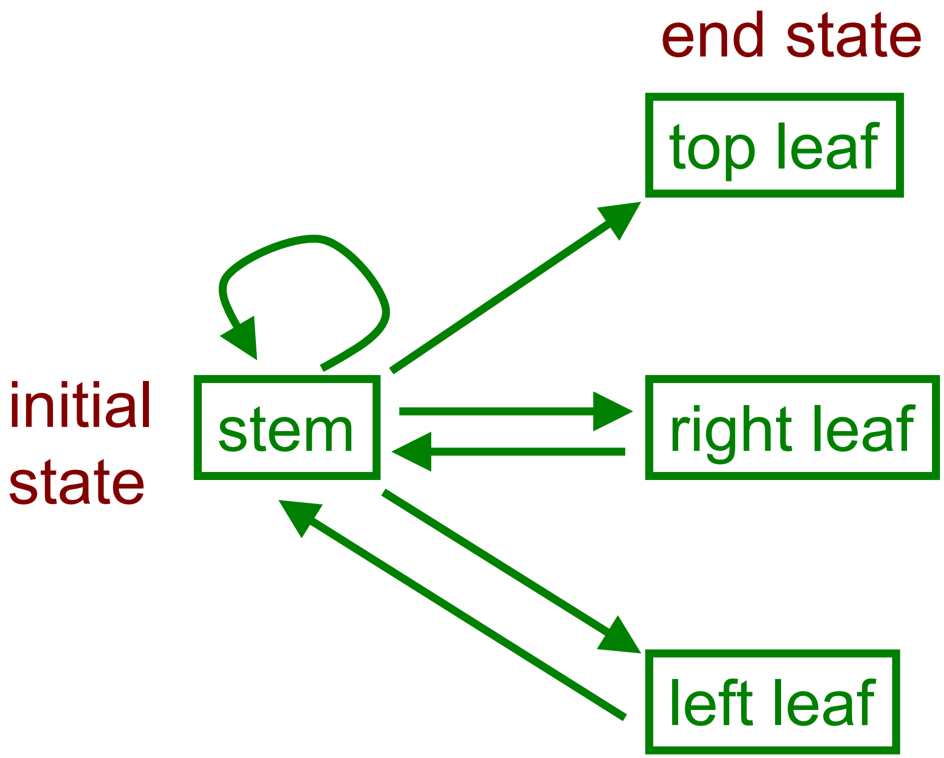
Generating a Branch

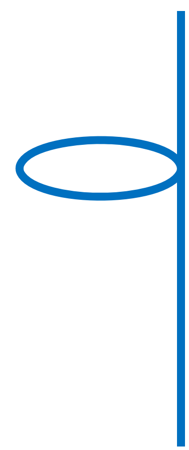


- Choose a legal initial state.
- Choose a shape from the state's shape distribution.



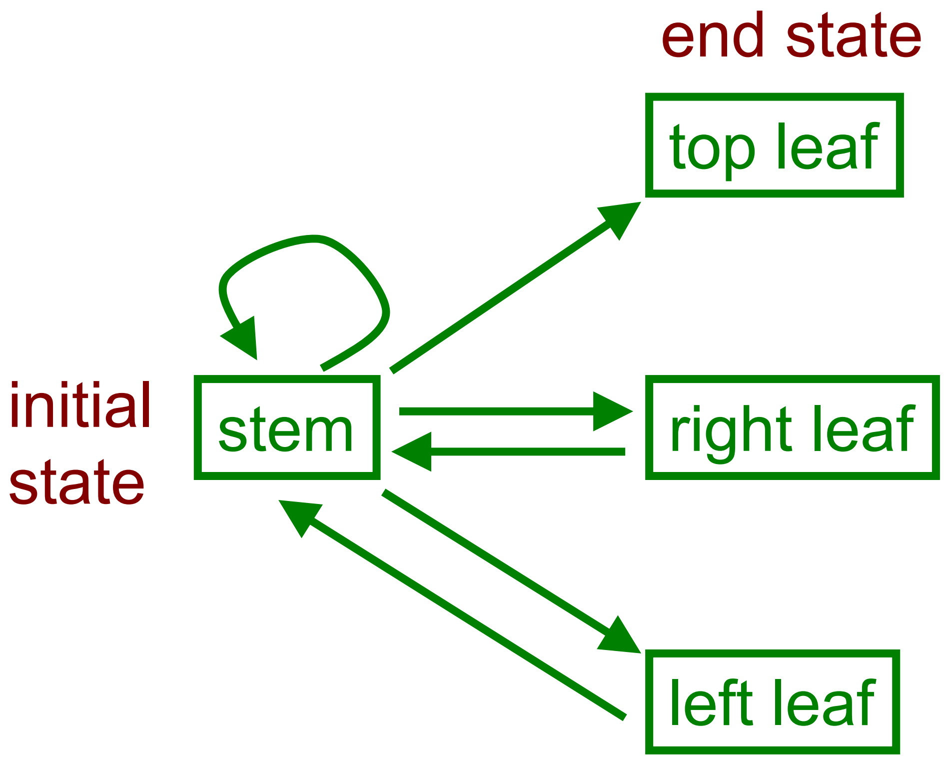
Generating a Branch



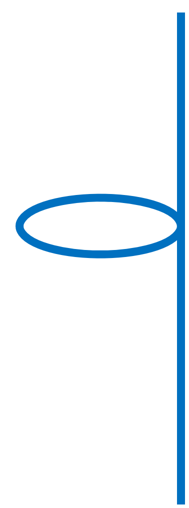


- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.

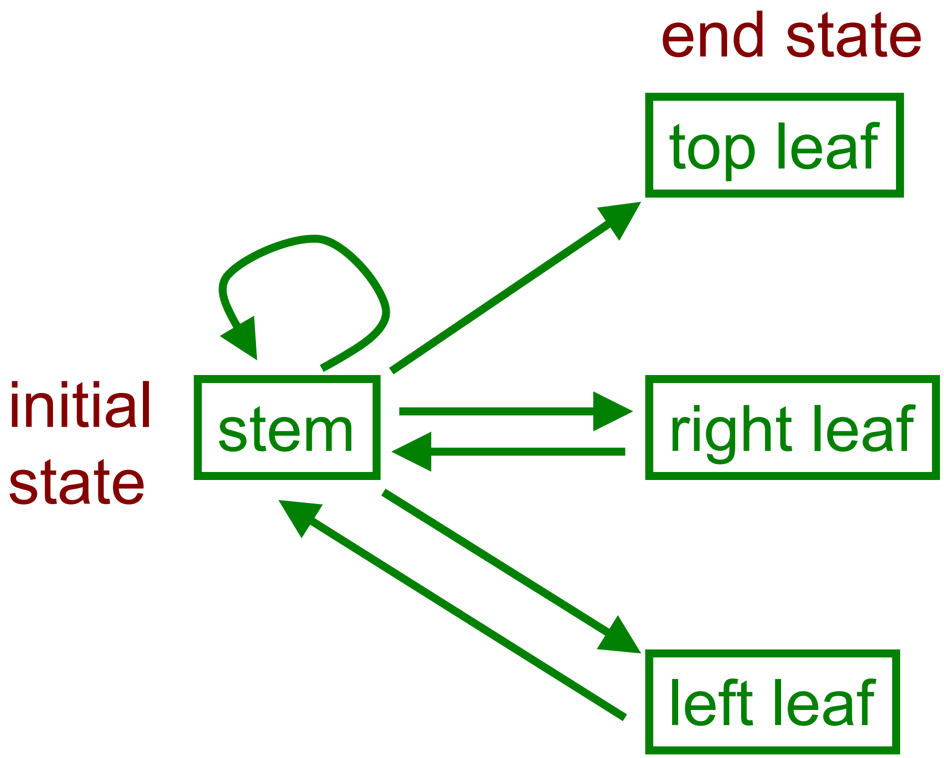
Generating a Branch

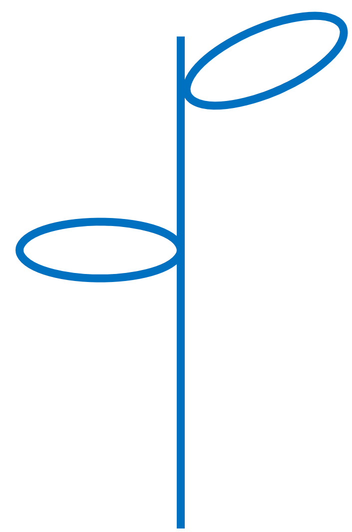


- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.



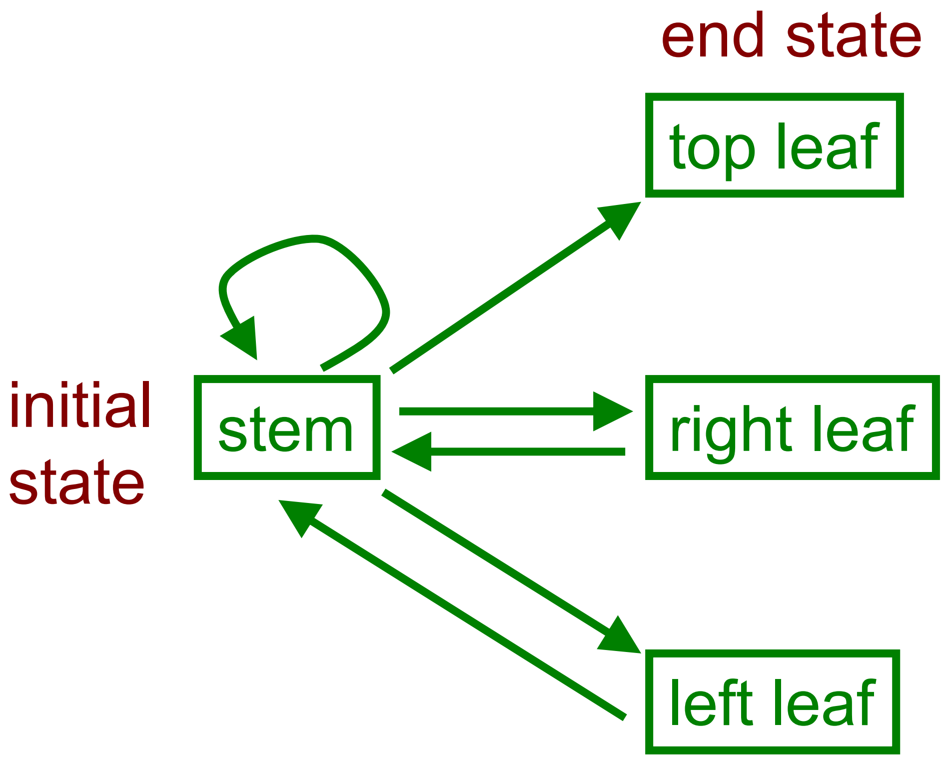
Generating a Branch



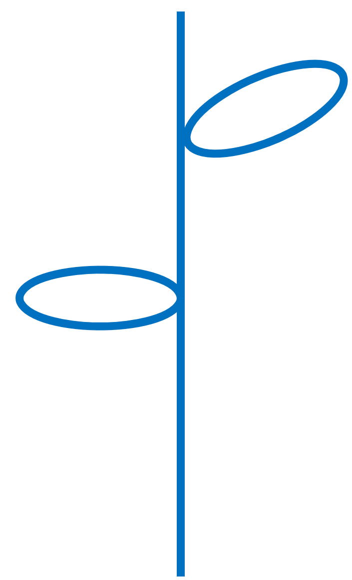


- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.

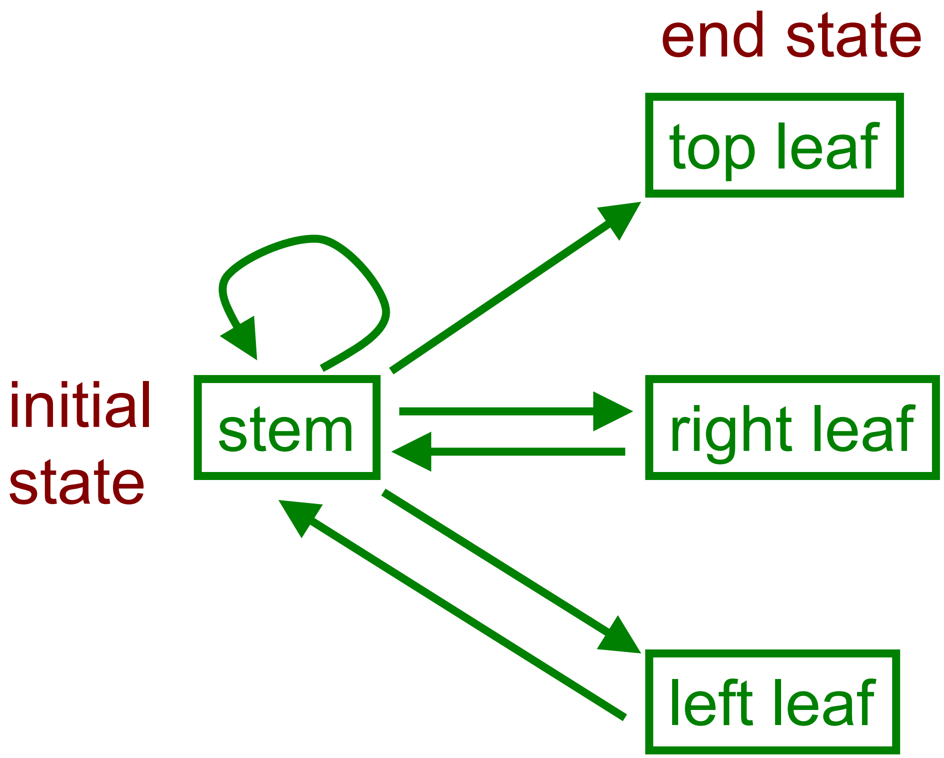
Generating a Branch

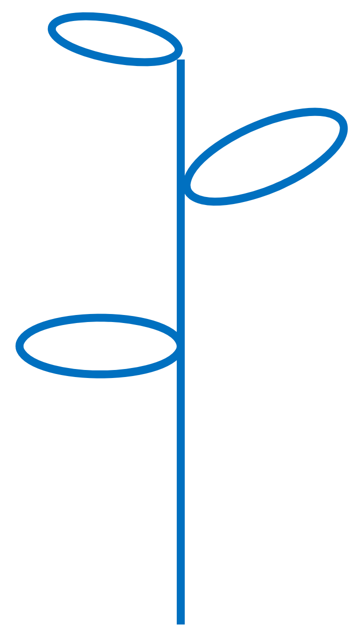


- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.



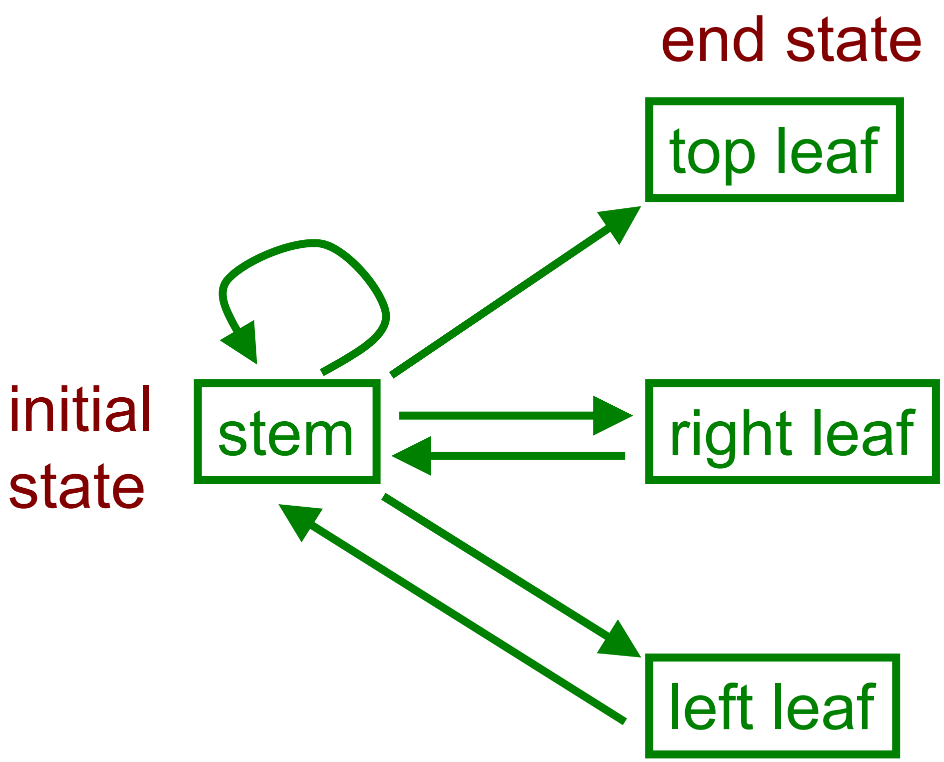
Generating a Branch

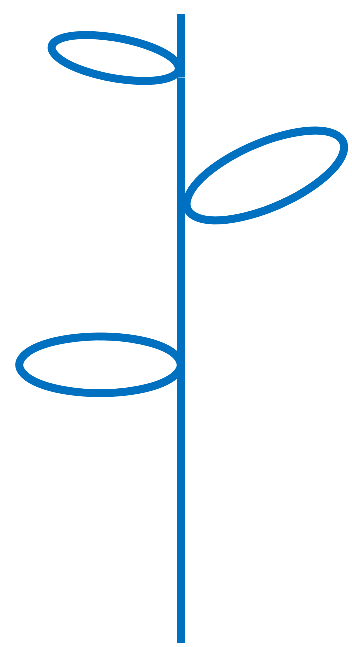




- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.

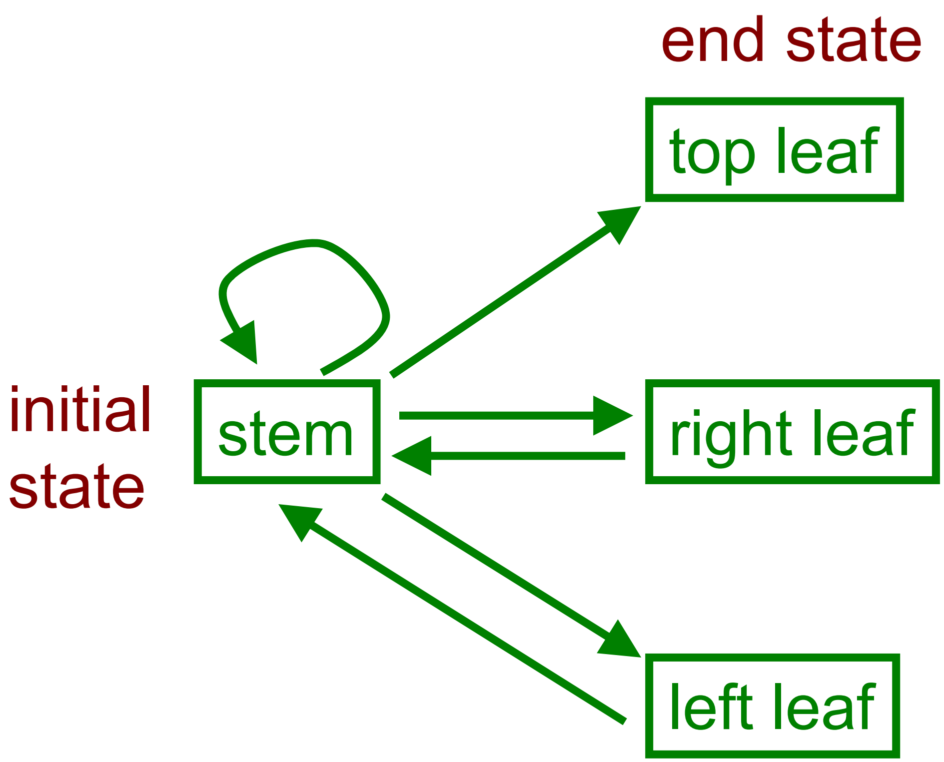
Generating a Branch

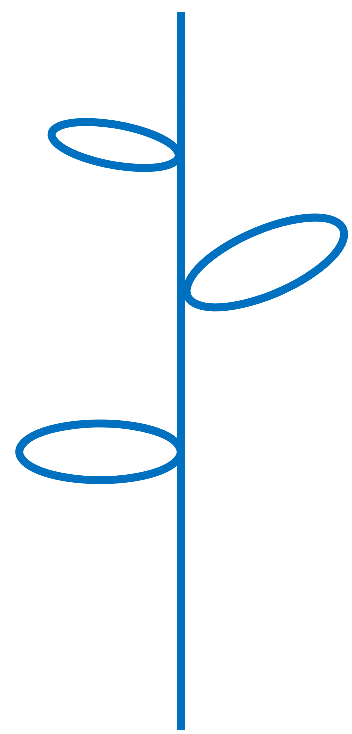




- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.

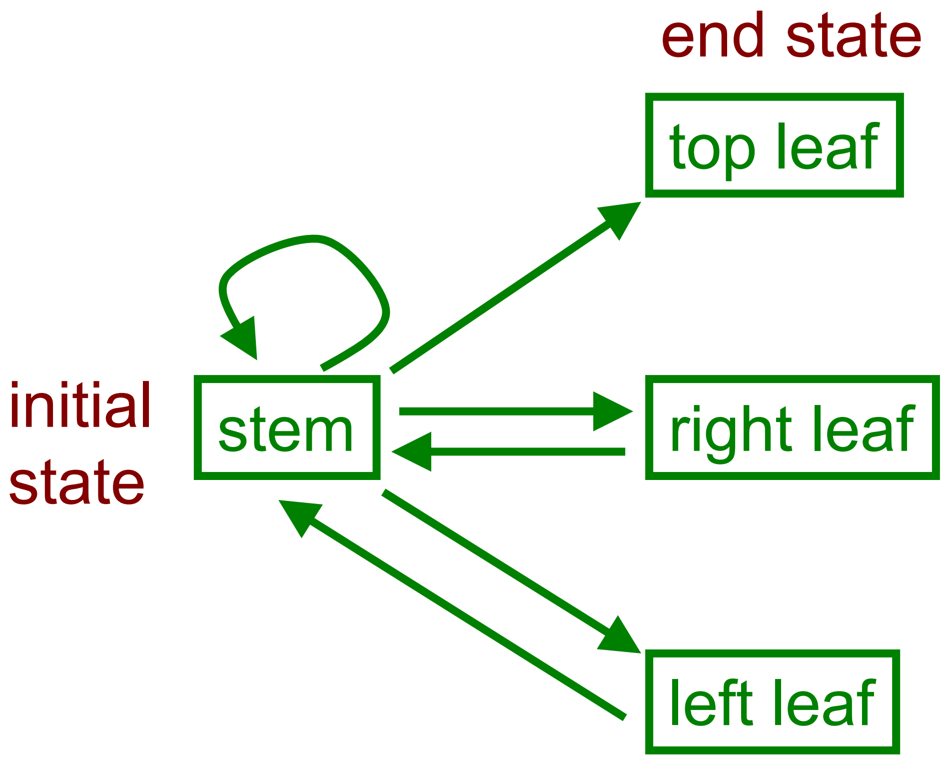
Generating a Branch

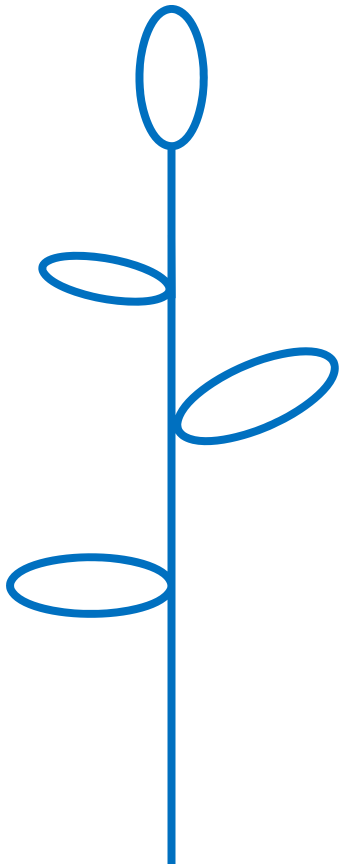




- Choose a legal transition to a next state.
- Choose a shape from the next state's shape distribution.

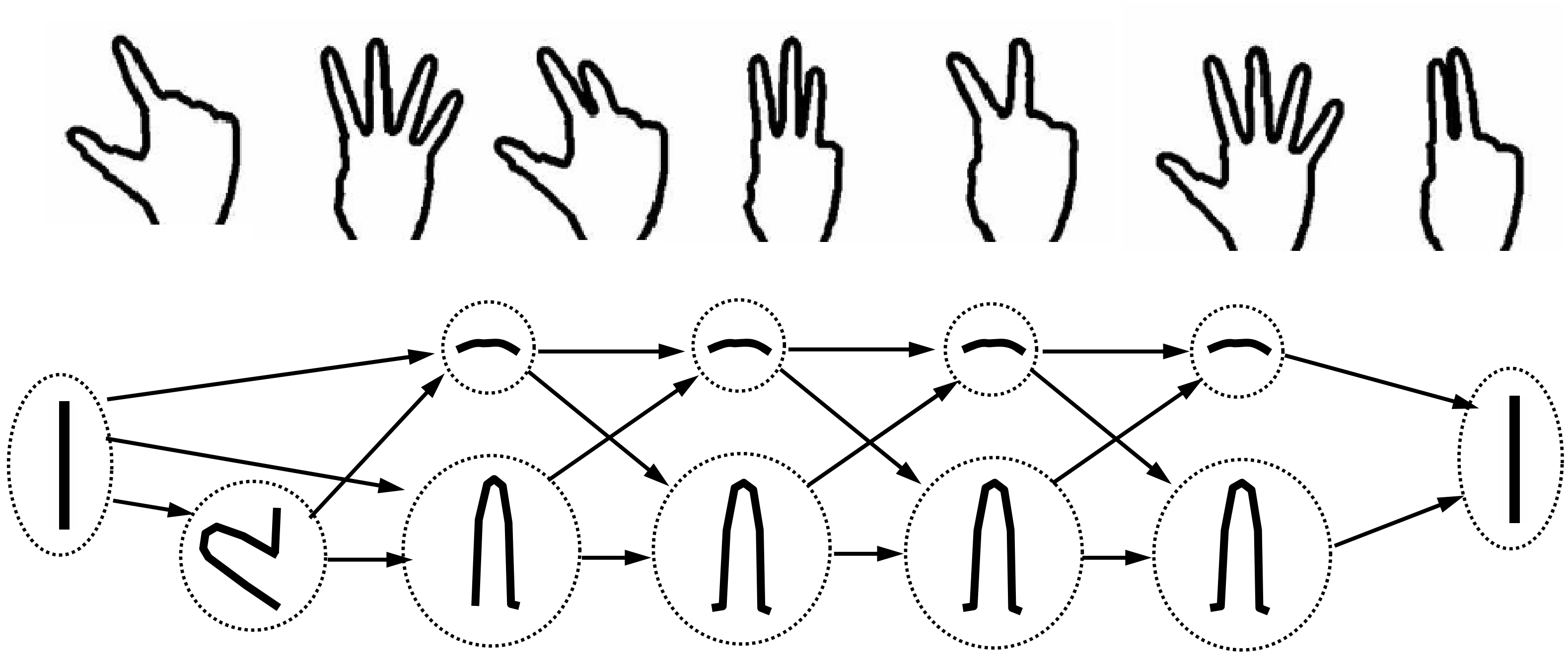
Generating a Branch





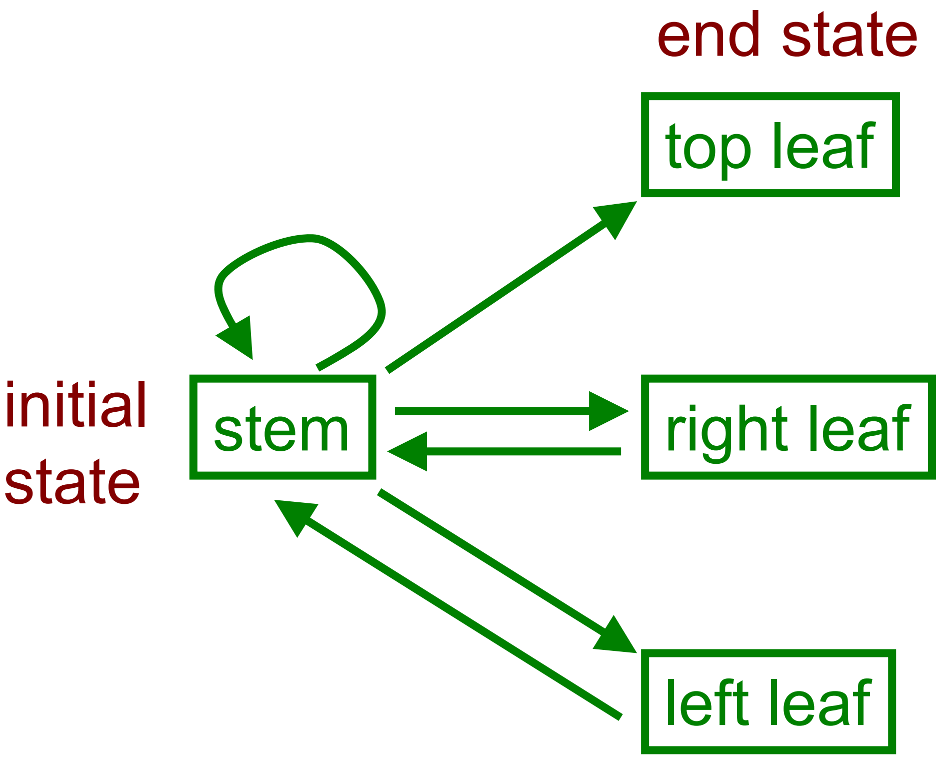
- Can stop when at a legal end state.

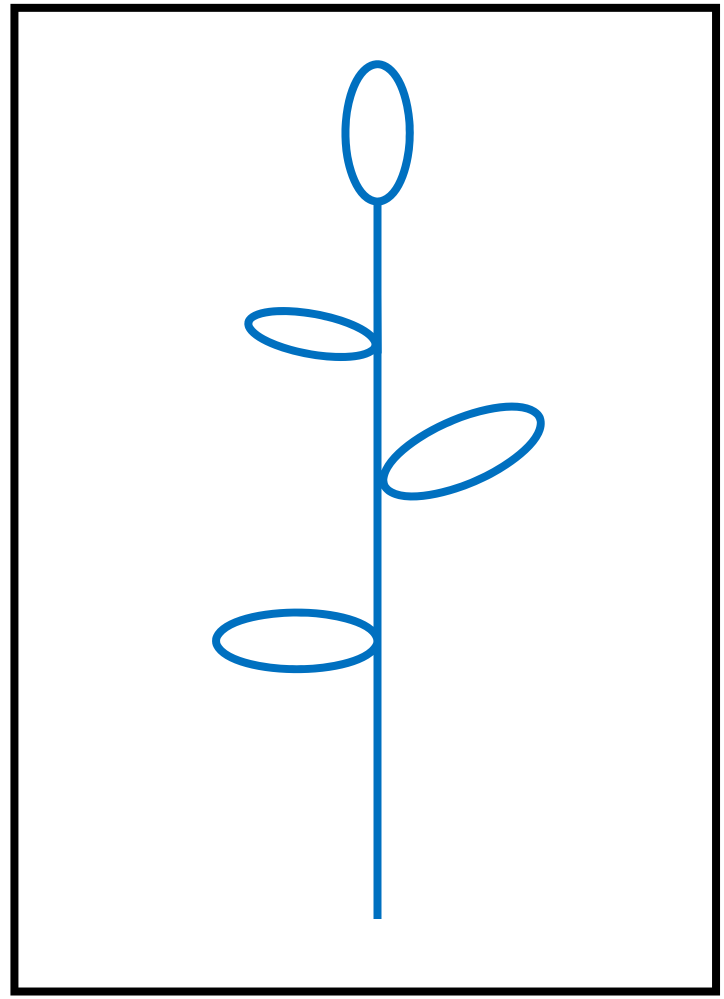
An HMM for Handshapes



- Each circle is a state (object part).
- Arrows show legal state transitions.

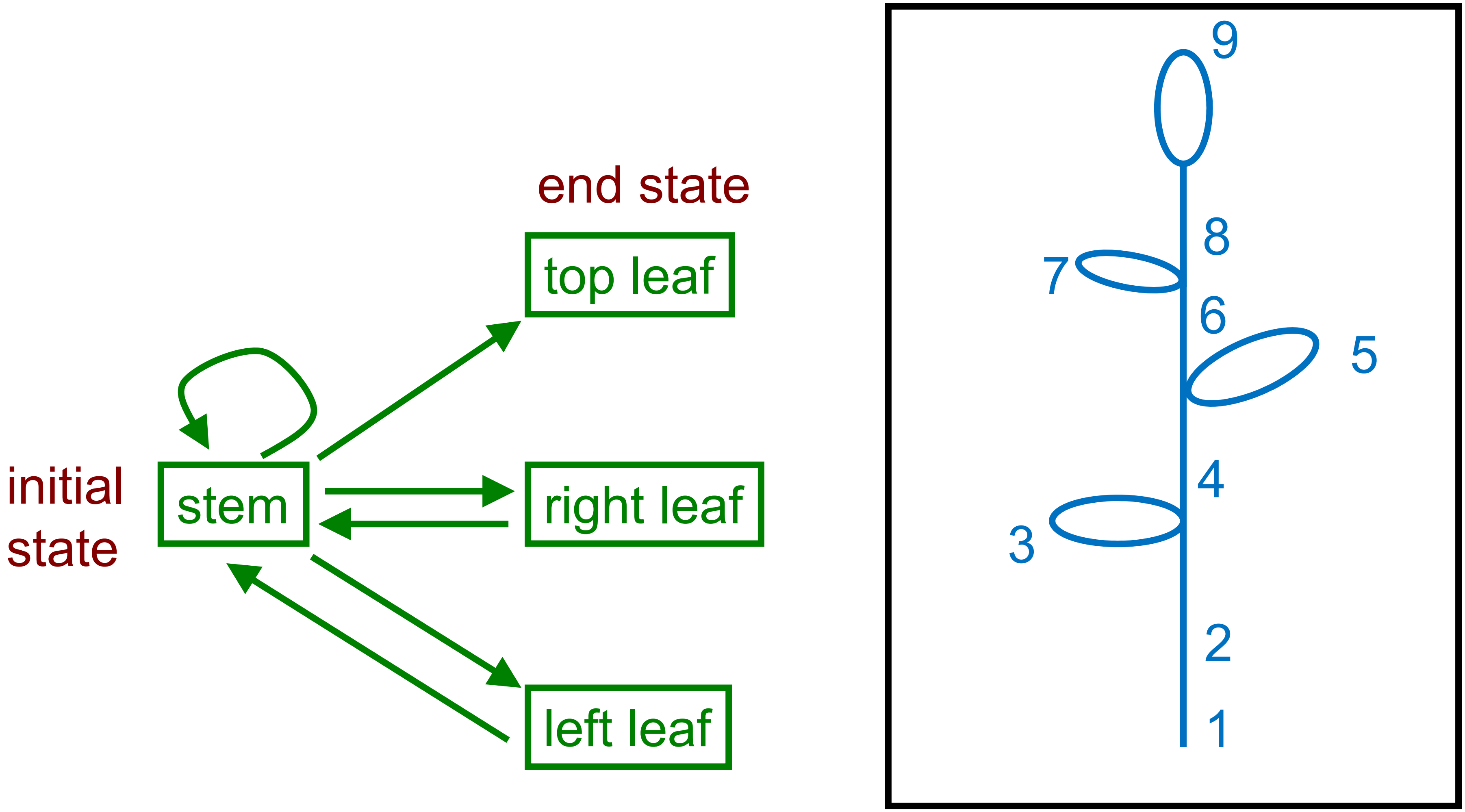
Matching Observations via DTW





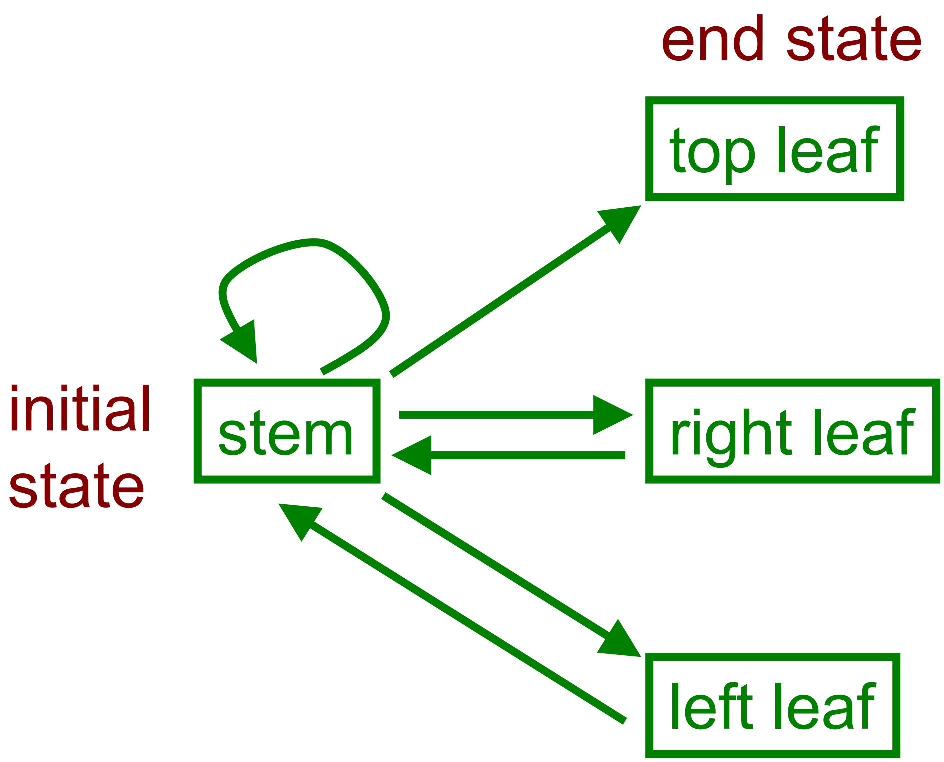
- Given a sequence of shape parts, what is the optimal sequence of corresponding states?

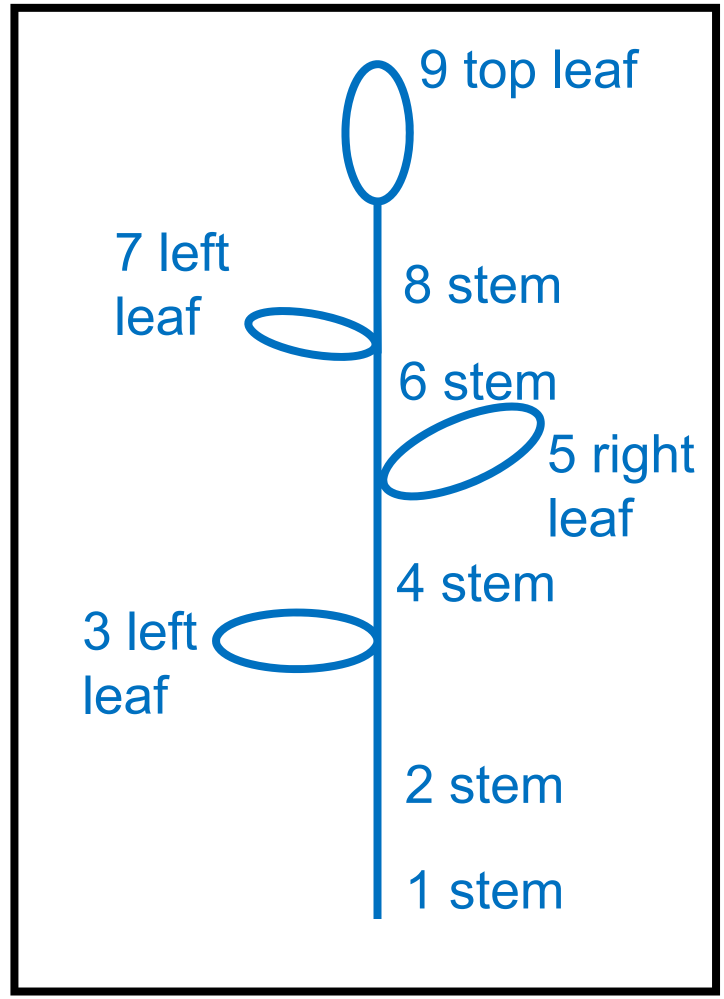
Matching Observations via DTW



- Given a sequence of shape parts, what is the optimal sequence of corresponding states?

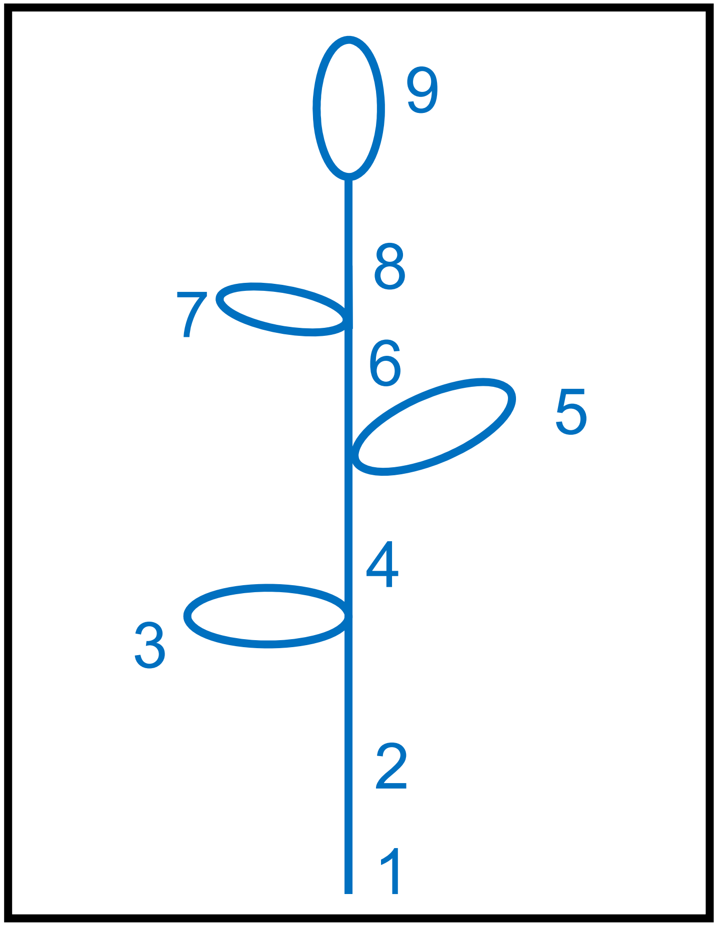
Matching Observations via DTW





- The Viterbi algorithm produces a globally optimal answer.

DTW/HMMs Cannot Handle Clutter



clean image, ordered observations → DTW/HMM can parse the observations

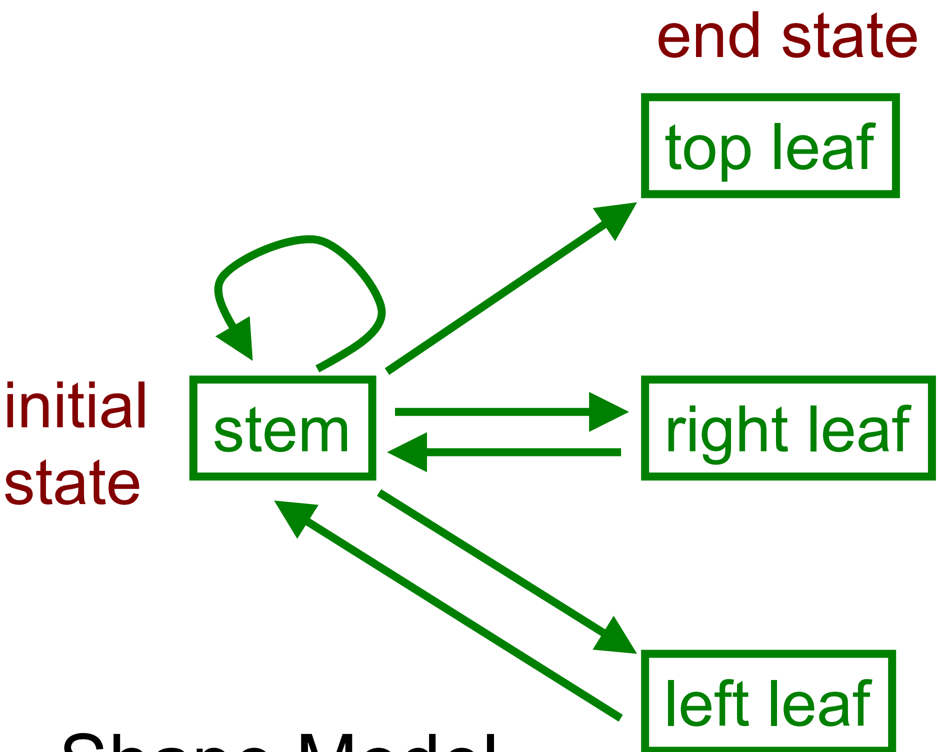
cluttered image, unordered observations

Key Challenges in Clutter

- The observations are not ordered.
- Many (possibly most) observations should not be matched.
- Solution: Hidden State Shape Models (HSSMs).
  - Extending HMMs and the Viterbi algorithm to address clutter.

Inputs



- Shape Model.
  - Shape parts.
  - Legal transitions.
  - Initial/end states.
  - More…

possible leaf locations

possible stem locations

- Possible matches for each model state.

Algorithm Input/Ouptut

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

possible leaf locations

possible stem locations

oriented edge pixels

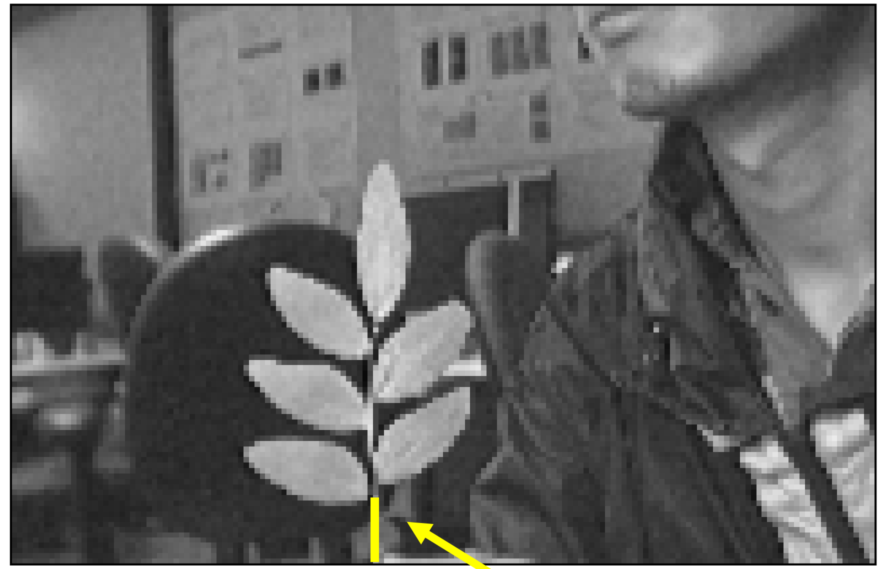
Algorithm Input/Output

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

Algorithm Input/Output

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.



- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

Algorithm Input/Output

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

Algorithm Input/Output

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

Q_3: stem

- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

Algorithm Input/Output

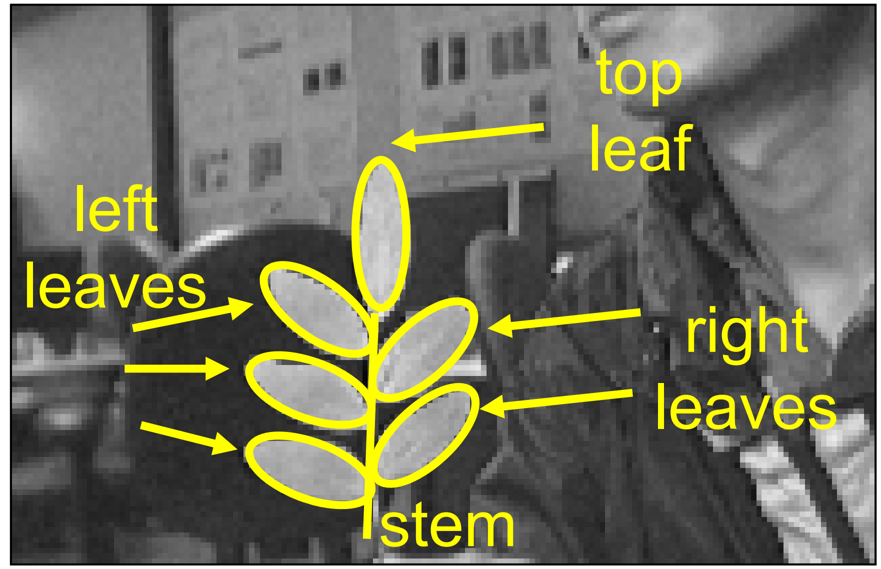
- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

Q_4: right leaf

- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

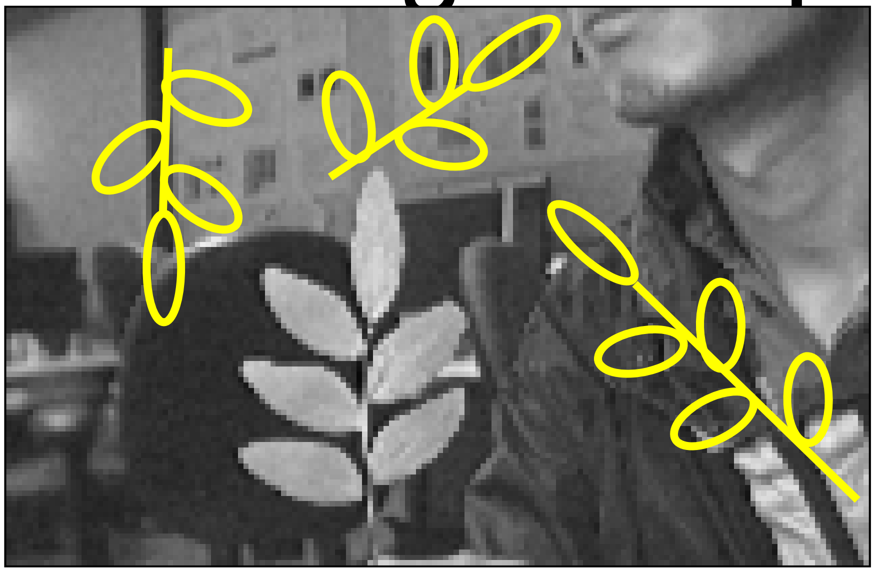
Algorithm Input/Output

- Input: Set of K image features $\{F_1, F_2, \ldots, F_K\}$.

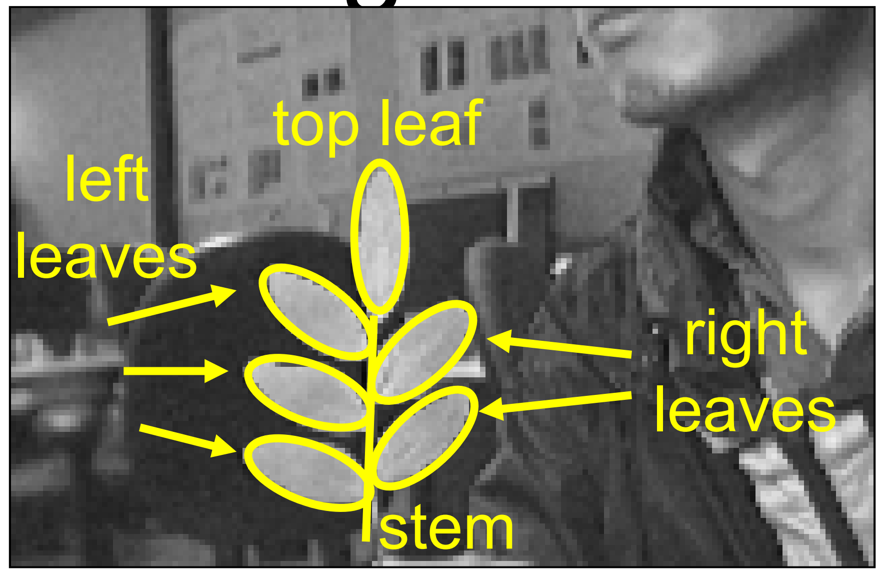


- Output: Registration.
  - $((Q_1, O_1), (Q_2, O_2), \ldots, (Q_T, O_T))$.
  - *Sequence* of matches. Q_j: model state. : O_j feature.
  - Model states tell us the *structure*.
  - Features tell us the *location*.

Finding the Optimal Registration



some possible registrations

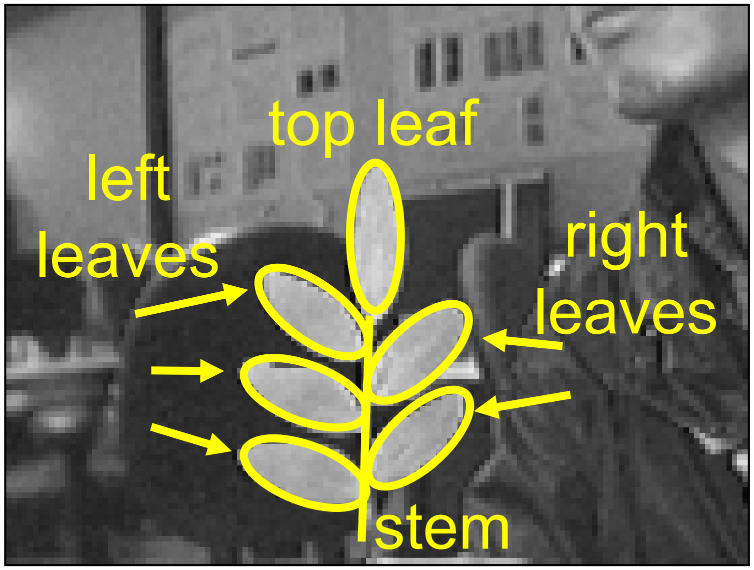


optimal registration

- Number of possible registrations is exponential to number of image features.
  - Evaluating each possible registration is intractable.
- We can find optimal registration in polynomial time (quadratic/linear to the number of features).
  - Using dynamic programming (modified Viterbi).

Evaluating a Registration

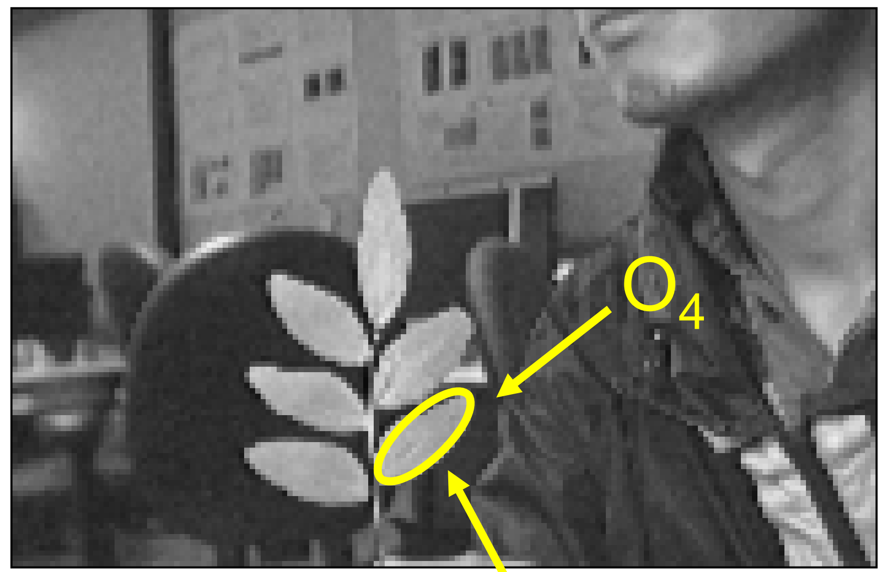
- Registration:
  - $((Q_1, O_1), (Q_2, O_2), (Q_3, O_3), (Q_4, O_4), \ldots, (Q_T, O_T))$.
- Probability is product of:
  - $I(Q_1)$: prob. of initial state.
  - $B(Q_j, O_j)$: prob. of feature given model state.
  - $A(Q_j, Q_{j+1})$: prob. of transition from Q_j to Q_{j+1}.
  - $D(Q_j, O_j, Q_{j+1}, O_{j+1})$: prob. of observing O_{j+1} given state = Q_{j+1}, and given previous pair (Q_j, O_j).
    - Not in HMMs, because there the order is known.



optimal registration

Finding the Optimal Registration

- Dynamic Programming algorithm:
  - Break up problem into smaller, interdependent problems.
- Definition: W(i, j, k) is the optimal registration such that:
  - Registration length is j.
  - $Q_j = S_i$.
  - $O_j = F_k$.
- Optimal registration:
  - W(i, j, k) with highest probability, such that Q_j is a legal end state.
- Suffices to compute all W(i, j, k).



Q_4: right leaf

Finding the Optimal Registration

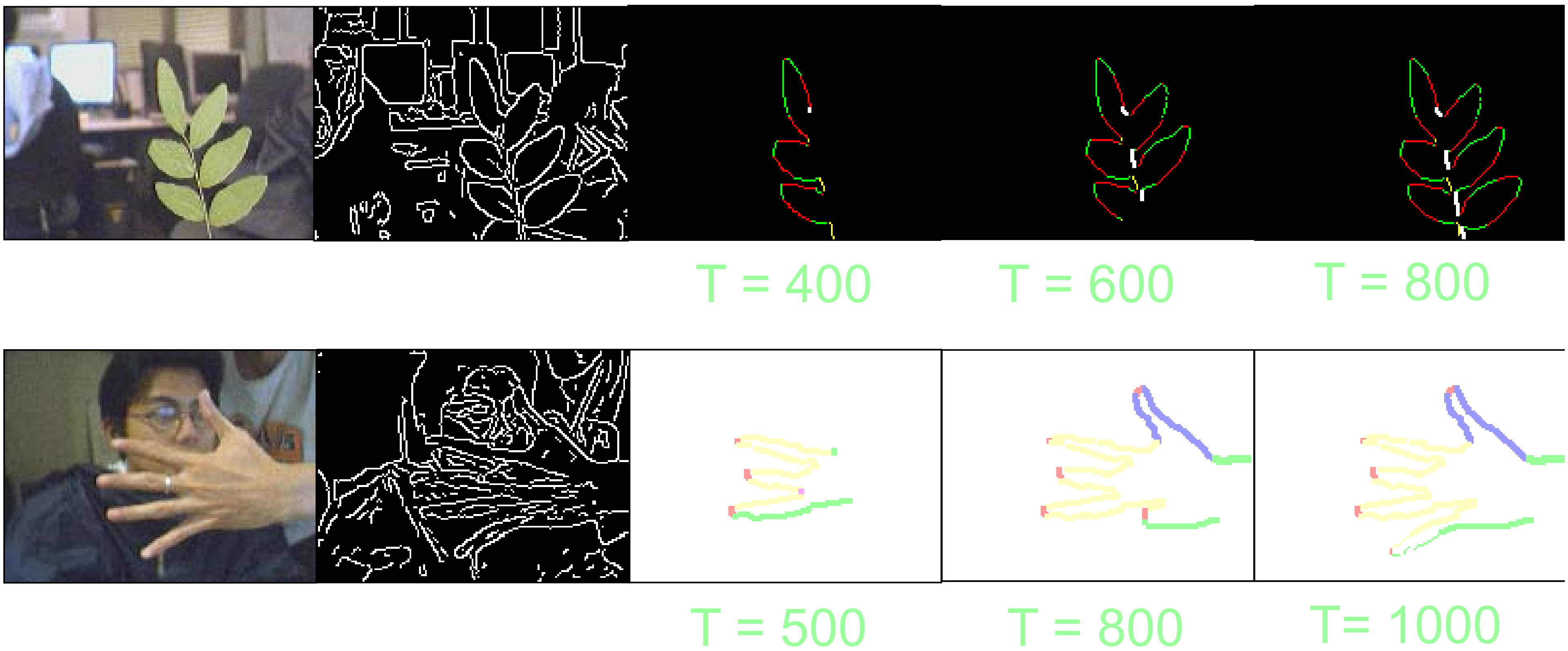
- $W(i, j, k)$: highest prob. registration such that:
  - Registration length is j.
  - $Q_j = S_i$.
  - $O_j = F_k$.
- $W(i, 1, k)$ is trivial: only one choice.
  - Registration consists of pairing S_i with F_k.
  - Cost: $I(S_i) + B(S_i, F_k)$:
- $W(i, 2, k)$ is non-trivial: we have choices.
  - Registration: $((Q_1, O_1), (Q_2, O_2))$
  - $Q_2 = S_i$, $O_2 = F_k$.
  - What we do not know: Q_1, O_1.
    - We can try all possible (Q_1, O_1), not too many.

Finding the Optimal Registration

- W(i, 3, k): even more choices.
  - Registration: $((Q_1, O_1), (Q_2, O_2), (Q_3, O_3))$
  - $Q_3 = S_i$, $O_3 = F_k$.
  - What we do not know: Q_1, O_1, Q_2, O_2.
- Here we use dynamic programming.
  - $W(i, 3, k) = ((Q_1, O_1), (Q_2, O_2), (Q_3, O_3))$ implies that $W(i', 2, k') = ((Q_1, O_1), (Q_2, O_2))$.
  - If we have computed all W(i, 2, k), the number of choices for W(i, 3, k) is manageable.
- This way we compute W(i, j, k) for all j.
  - The number of choices does no increase with j.

Unknown-scale Problem

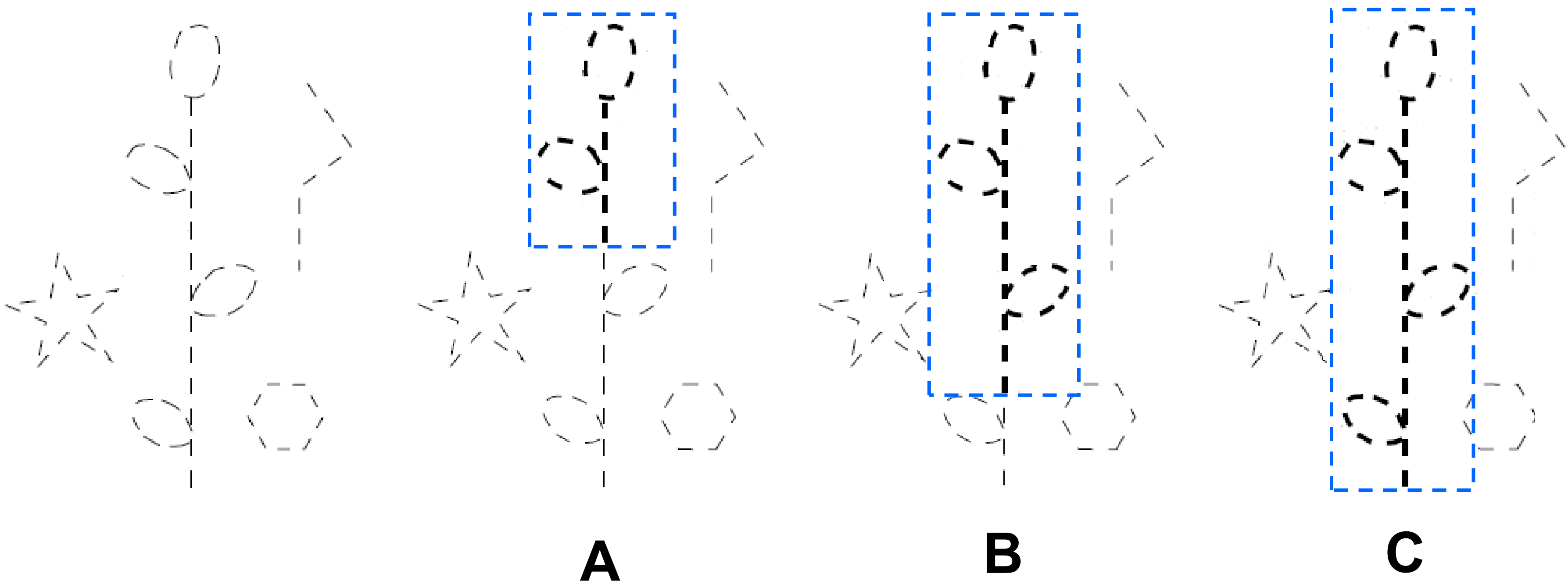
What is the length of the optimal registration?



- Probability decreases with registration length.
  - HMMs are biased towards short registrations

Unknown-scale Problem

What is the length of the optimal registration?

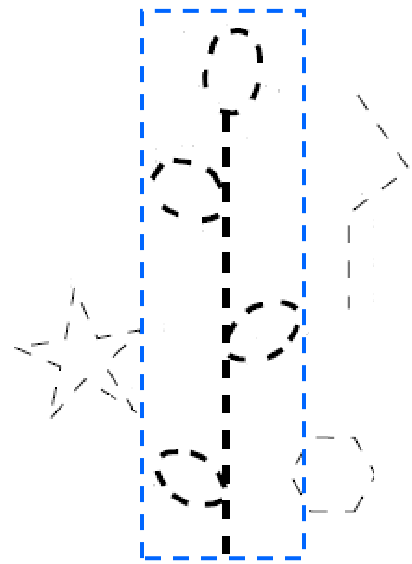


- Probability decreases with registration length.
  - HMMs are biased towards short registrations

Handling Unknown Scale

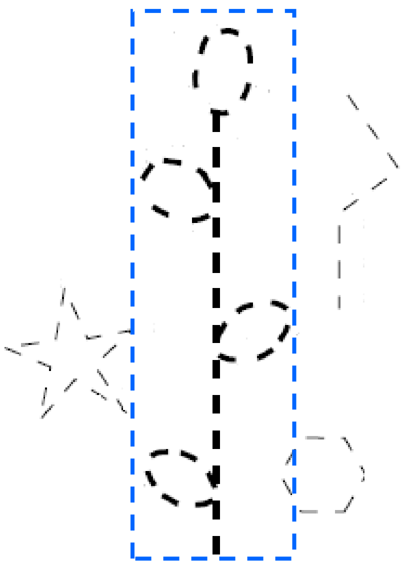
- Registration:
  - $((Q_1, O_1), (Q_2, O_2), (Q_3, O_3), (Q_4, O_4), \ldots, (Q_T, O_T))$.

- Probability is product of:
  - $I(Q_1)$: prob. of initial state.
  - $B(Q_j, O_j)$: prob. of feature given model state.
  - $A(Q_j, Q_{j+1})$: prob. of transition from Q_j to Q_{j+1}.
  - $D(Q_j, O_j, Q_{j+1}, O_{j+1})$: prob. of observing O_{j+1} given state = Q_{j+1}, and given previous pair (Q_j, O_j).



Handling Unknown Scale

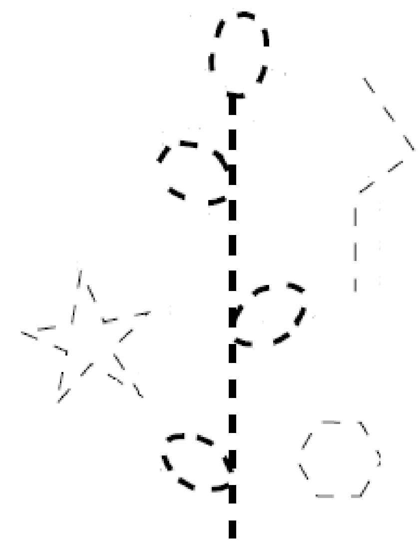
- Registration:
  - $((Q_1, O_1), (Q_2, O_2), (Q_3, O_3), (Q_4, O_4), \ldots, (Q_T, O_T))$.
- Probability is product of:
  - $I(Q_1)$: prob. of initial state.
  - $B(Q_j, O_j)$: prob. of feature given model state.
  - $A(Q_j, Q_{j+1})$: prob. of transition from Q_j to Q_{j+1}.
  - $D(Q_j, O_j, Q_{j+1}, O_{j+1})$: prob. of observing O_{j+1} given state = Q_{j+1}, and given previous pair (Q_j, O_j).



Handling Unknown Scale

- Before: $B(Q_j, O_j)$: prob. of feature given state.
  - Adding a feature decreases registration probability.
- Now: $B(Q_j, O_j)$ is a ratio:

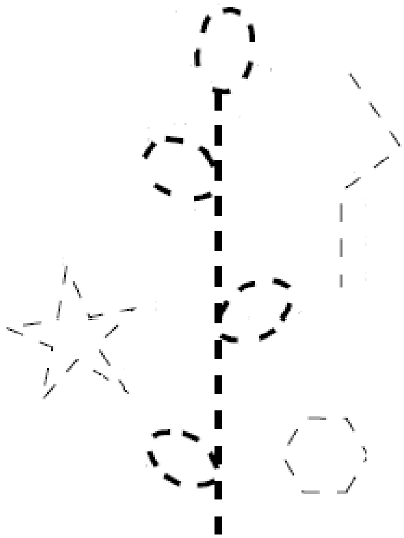
$$\frac{P(O_j \mid Q_j)}{P(O_j \mid \text{clutter})}$$



  - $B(Q_j, O_j)$ can be greater or less than 1.
  - Adding a feature may increase or decrease registration probability.
  - Bias towards short registrations is removed.

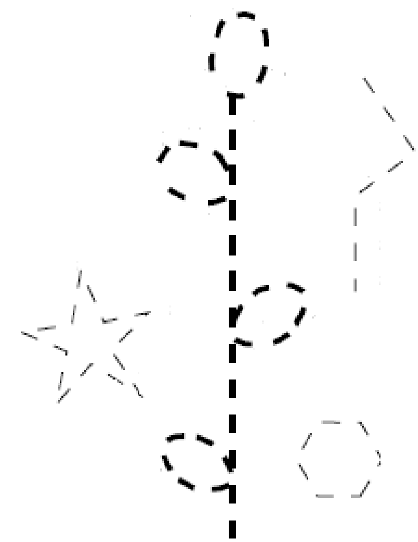
Another Perspective

- Before: probability depended only on features matched with the model.
  - Fewer features → higher probability.
- The probability function should consider the entire image.
  - Features matched with model.
  - Features assigned to clutter.



Another Perspective

- For every feature, does it match better with the model or with clutter?
  – Compute $P(F_k \mid \text{clutter})$.
- Given a registration R, define:
  – C(R): set of features left out.
  – F: set of all image features.
- Total probability:
  – P(registration) P(C(R) | clutter)).



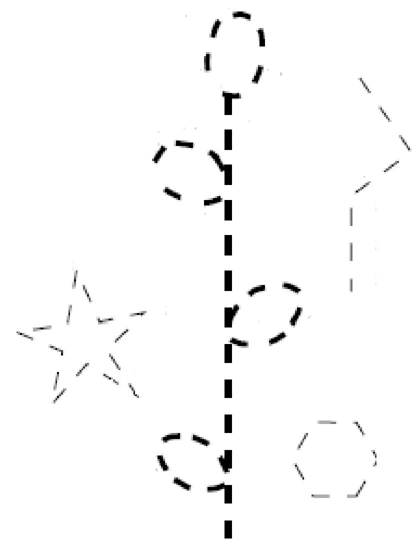
C(R): gray
F: gray/black

Another Perspective

- Given a registration R, define:
  - C(R): set of features left out.
  - F: set of all image features.
- Total probability:
  - P(registration) P(C(R) | clutter))

  proportional to:

$$\frac{P(\text{registration})\ P(C(R) \mid \text{clutter}))}{P(F \mid \text{clutter}))}$$

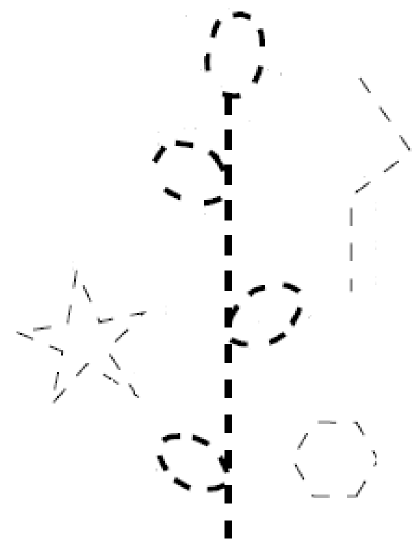


C(R): gray
F: gray/black

Another Perspective

- Goal: maximize $\dfrac{P(\text{registration})\ P(C(R) \mid \text{clutter}))}{P(F \mid \text{clutter}))}$

- Equal to product of:
  - $I(Q_1)$: prob. of initial state.
  - $B(Q_j, O_j)$: $\dfrac{P(O_j \mid Q_j)}{P(O_j \mid \text{clutter})}$
  - $A(Q_j, Q_{j+1})$: prob. of transition from Q_j to Q_{j+1}.
  - $D(Q_j, O_j, Q_{j+1}, O_{j+1})$: prob. of observing O_j given state = Q_j, and given previous pair (Q_{j+1}, O_{j+1}).



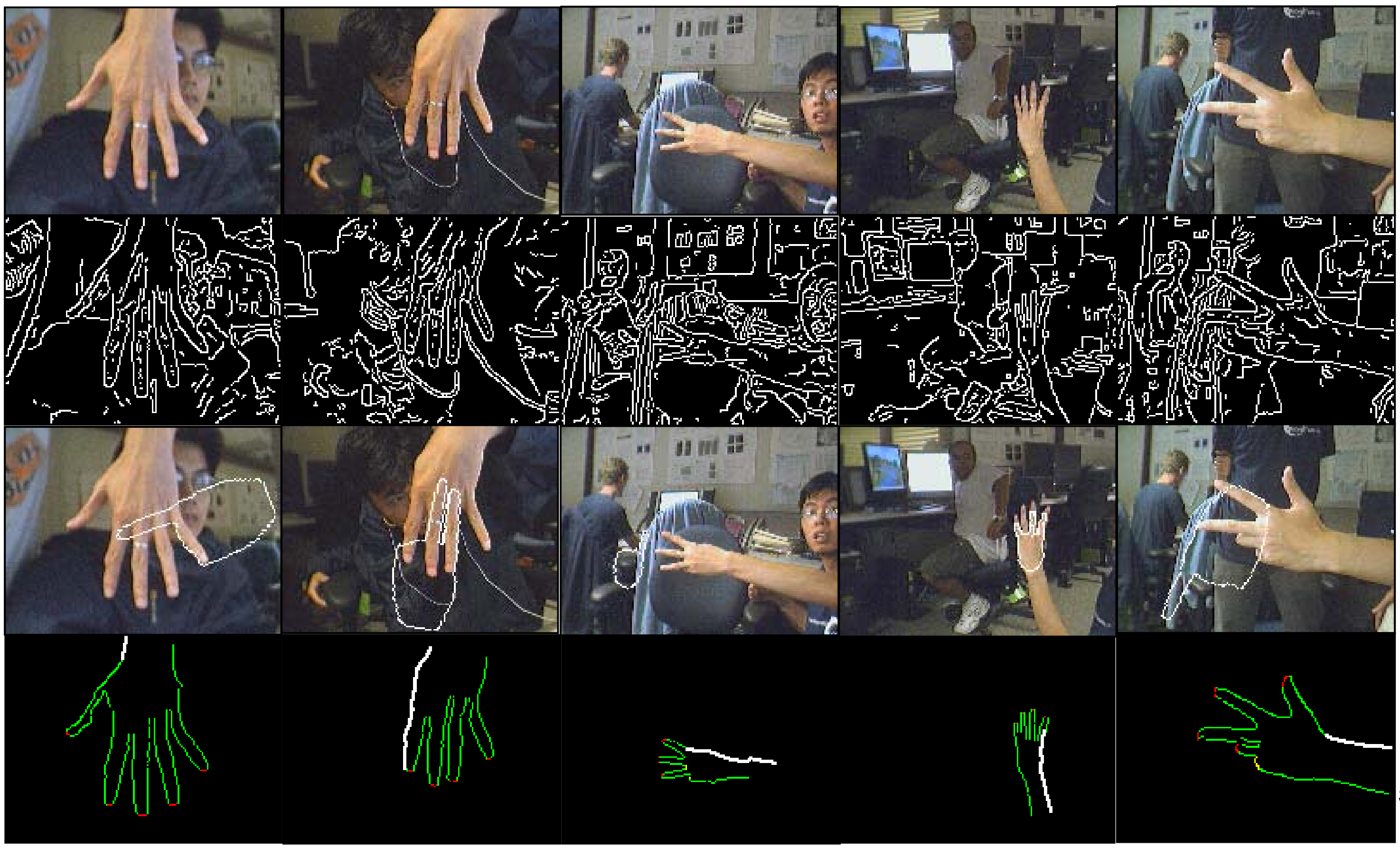
C(R): gray
F: gray/black

Experiments

- Datasets:
  - 353 hand images.
  - 100 images of branches.
- Methods.
  - HSSM: known scale.
  - HSSM+CM: unknown scale, with clutter modeling.
  - HSSM+CM+SEG: unknown scale, with clutter modeling + segmental model.
- Time: 8 different orientations.
  - 2-3 minutes per images for HSSM and HSSM+CM.
  - <30 minutes per images for HSSM+CM+SEG.

Results on Hands

	Chamfer	Oriented Chamfer	HSSM
Correct recognition	4%	22%	34%
Correct localization	35%	55%	60%

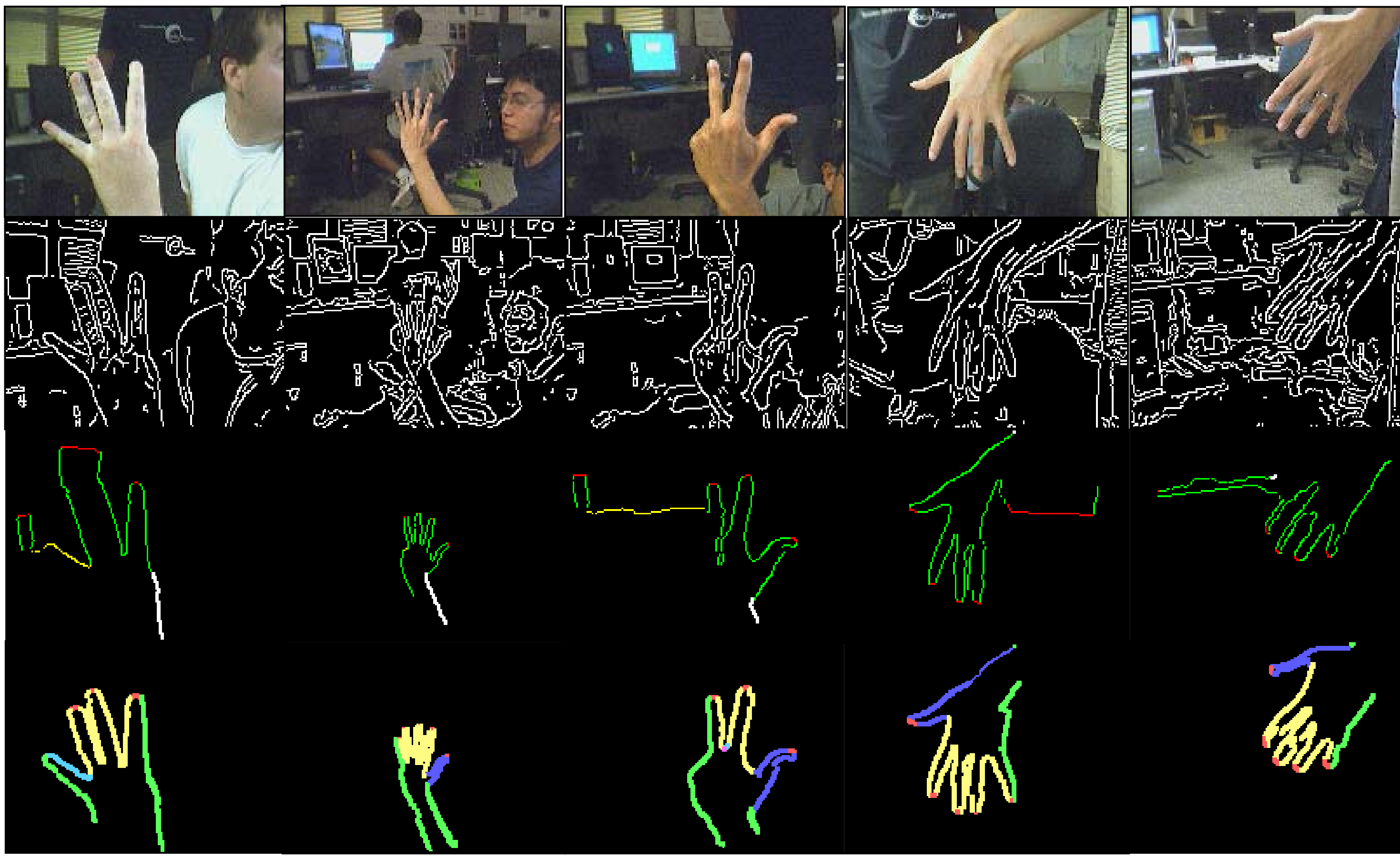


Chamfer Matching

HSSM

Results on Hands

	HSSM	HSSM+CM	HSSM+CM+SEG
Correct recognition	34%	59%	70%
Correct localization	60%	85%	95%



HSSM

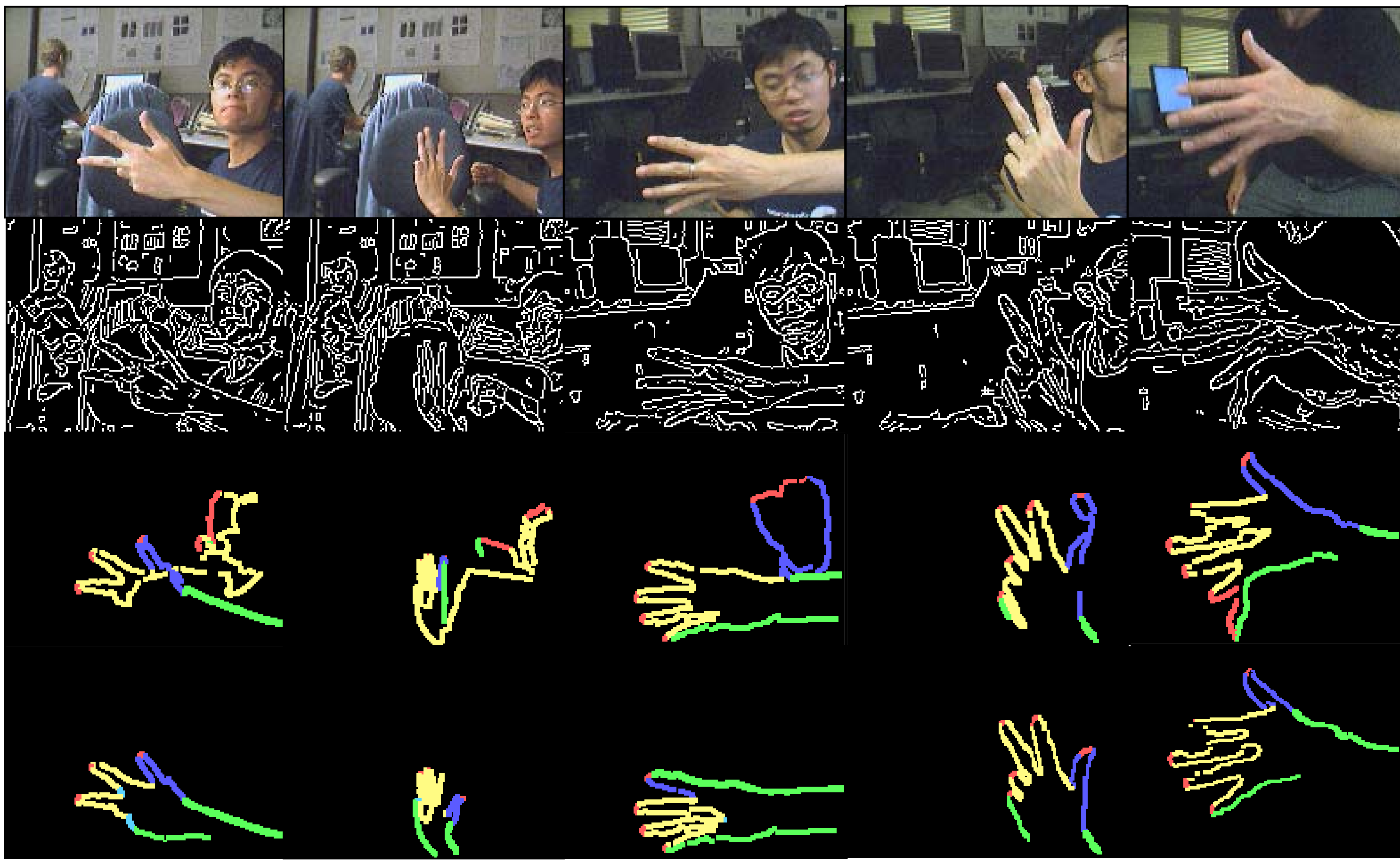
HSSM +CM

Results on Hands

	HSSM	HSSM+CM	HSSM+CM+SEG
Correct recognition	34%	59%	70%
Correct localization	60%	85%	95%

HSSM
+ CM

HSSM
+CM+SEG



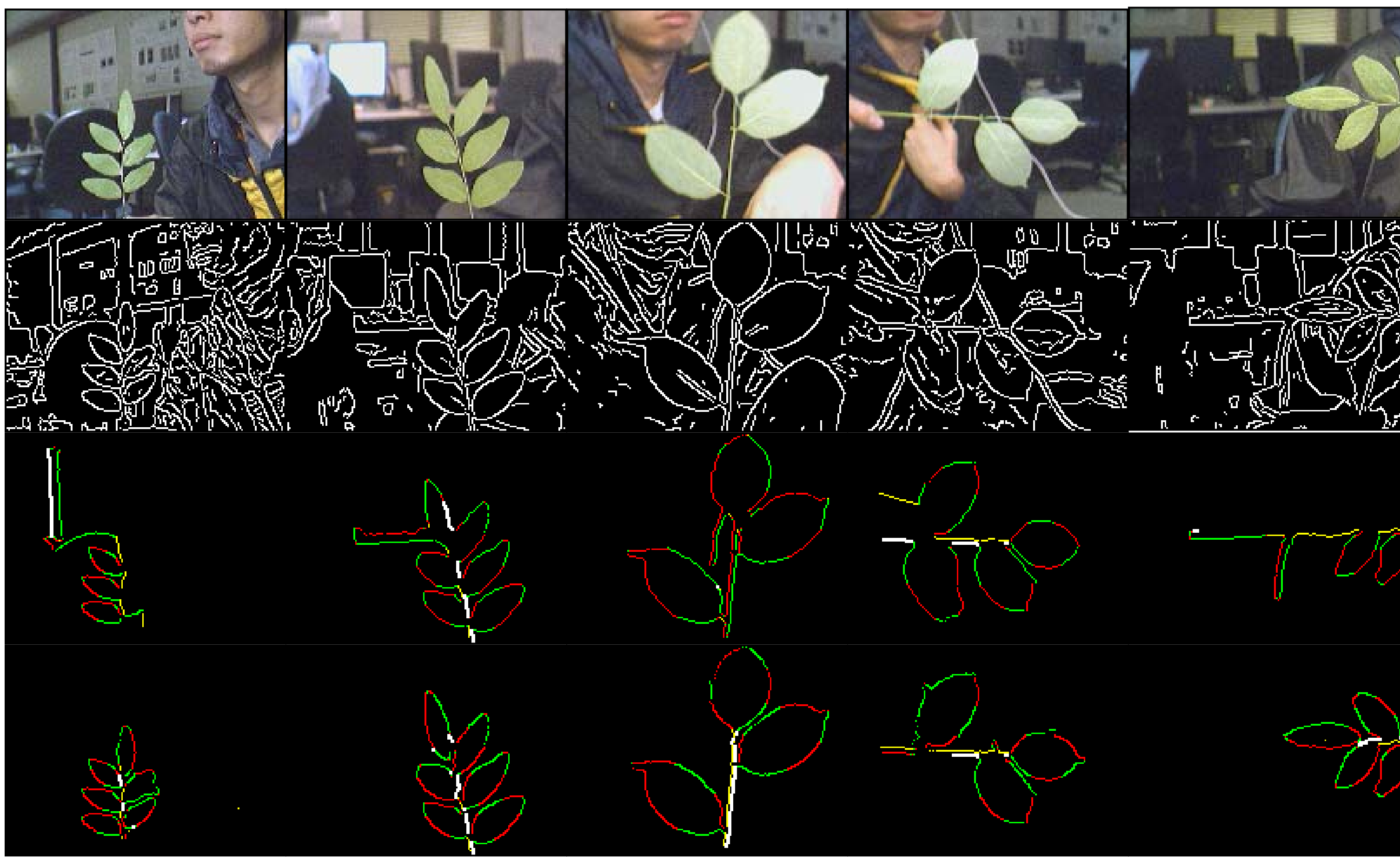
Results on Branches

	HSSM
Correct recognition	43%
Correct localization	79%

correct recognition with HSSM

Results on Branches

	HSSM	HSSM+CM	HSSM+CM+SEG
Correct recognition	43%	65%	N/A
Correct localization	79%	98%	N/A



HSSM

HSSM +CM

Recap

- HSSMs model objects of variable shape structure.
- Key advantages:
  - A single variable-shape model, vs. one model for each shape structure (exponential worst-case complexity).
  - Detection can tolerate very large amounts of clutter.
    - Object features can be a very small fraction of all features.
  - Optimal registration length found automatically.
  - Scale dependency between parts is captured using a segmental model.