Machines seeing Action

Aaron Bobick
School of Engineering and Applied Science
Washington University in St. Louis
afb@wust.edu
Washington University in St. Louis

A medium-sized, independent university located in the nation’s center, Washington University is counted among the world’s leaders in teaching, research, patient care and service to society.

Nearly 60 of the university’s graduate and undergraduate programs hold a top 25 ranking by *U.S. News & World Report*, including The Brown School of Social Work, ranked No. 1, and the School of Medicine, ranked No. 6.
WashU Engineering

Academic departments:
• Biomedical Engineering
• Computer Science & Engineering
• Electrical & Systems Engineering
• Energy, Environmental & Chemical Engineering
• Mechanical Engineering & Materials Science

Engineering students:
1,271 undergraduates
760 master’s
356 doctoral

$150 million invested in new and renovated engineering labs and classrooms

Washington University School of Engineering & Applied Science

engineering.wustl.edu
Activity recognition? or Seeing Action?

• What does it mean for a Machine to “See” and Activity or an Action?
• What does it mean for a computer to “See” anything?
• A simple example that we now all expect...
Seeing by machine...

http://www.faceplusplus.com/demo-detect/
Seeing? Action?

- “Seeing” is essentially making a decision about an image.
- Not “images in, images out” which is most image software. Also not geometry – or even simple pixel tricks.

(Sportvision 1st down line)

- But, “images in, decisions (labels) out”
Activity recognition

- But what about “Action” or “Activity”
- Informally: “Describe to me what’s happening.”
- Want to detect events, actions, incidents, activities...
  - Gait and gesture recognition
  - Video annotation – generate labels
  - Driving behavior – combination of recognition and prediction
  - Surveillance – allowed or dis-allowed activities
  - Perceiving robots interacting with people
Different levels of “action”...

- **Movement** - atomic behaviors defined by motion
  - "Bending down", “Swinging a hammer”
- **Action** – a single, semantically meaningful "event"
  - "Opening a door", "Lifting a package"
  - Typically short in time
  - Might be definable in terms of motion; especially so in a particular context.
- **Activity** – a behavior or collection of actions with a purpose/intention.
  - "Delivering packages“, “parking a car”
  - Typically has causal underpinnings
  - Can be thought of as statistically structured events
Different possible tasks

Task might require

• Class of event/activity *detection* – given some data relevant description of an event detect (label) that it has occurred. *Or lots of data*...

• Event/activity *recognition* – given a semantic description of some set of events, tell me which one happens. *Maybe not lots of data...but may be...*

• Event/activity *understanding* – be able to see a variation on an event, or a failed event. Think of “parsing” a video of an activity
Really old work: Strict Appearance

• Is recognizing movement a 3D or 2D problem? Simple human psychophysics and computational complexity argue for 2D aspects.

• *Temporal templates*: Movements are recognized directly from the motion.

• Appearance-based recognition can assist geometric recovery: recognition labels the parts and allows extraction.
Blurry Video
Less Blurry Video!
Shape and motion: view-based

- Schematic representation of sitting at 90°
Motion energy images

- Spatial accumulation of motion.
- Collapse over specific time window.
- Motion measurement method not critical (e.g. motion differencing).
Motion history images

- Motion history images are a different function of temporal volume.
- Pixel operator is replacement decay:

\[
\begin{align*}
\text{if moving } & \ I_\tau(x,y,t) = \tau \\
\text{otherwise } & \ I_\tau(x,y,t) = \max(I_\tau(x,y,t-1)-1,0)
\end{align*}
\]

- Trivial to construct \( I_{\tau-k}(x,y,t) \) from \( I_\tau(x,y,t) \) so can process multiple time window lengths without more search.
- MEI is thresholded MHI
Temporal-templates

- $\text{MEI} + \text{MHI} = \text{Temporal template}$
Aerobics examples
Recognizing temporal templates
(PAMI 2001, Bobick and Davis)

- For MEI and MHI compute global properties (e.g. Hu moments). Treat both as grayscale images.
- Collect statistics on distribution of those properties over people for each movement.
- At run time, construct MEIs and MHIs backwards in time.
  - Recognizing movements as soon as they complete.
- Linear time scaling.
  - Compute range of $\tau$ using the min and max of training data.
- Simple recursive formulation therefore very fast.
- Filter implementation obvious so biologically “relevant”.
The KidsRoom

• A narrative, interactive children’s playspace.

• Demonstrates computer vision “action” recognition.

• Sometimes, possible because the machine knows the context.

• Ported to the Millennium Dome, London, 2001
Recognizing Movement in the KidsRoom

- First teach the kids, then observe.
- Temporal templates “plus”
- Monsters always do something, *but only speak it when sure.*
Lesson from Temporal Templates:

*It's all about the representation ...*
Data-driven vs Knowledge-taught

Data-driven

Statistical

Knowledge

Structural

Movement

Temporal and relational complexity

Activity

MHI's

PHMM's

P-Net's

SCFG's

Suffix Trees

Action BN's

PNF

MACHINE LEARNING (> 2000)
Data-driven vs Knowledge-taught

Data-driven
- Statistical

Knowledge
- Structural

Movement
- Temporal and relational complexity
- Activity

MACHINES LEARNING (> 2000)

PHMM's
- P-Net's
- SIN's
- SCFG's
- PNF
- Suffix Trees
- Action BN's
- Event BN's
- MHI's
Older Structured Representations for Activity Recognition

• Parametric HMMs for “structured” gesture recognition
  • Capture the systematic variation of parameterized gesture for *recognition*.

• Grammar-based representation and parsing
  • Highly expressive for activity description
  • Easy to build higher level activity from reused low level vocabulary.

• P-Net (Propagation nets) – really stochastic Petri nets
  • Specify the structure – with some annotation can learn detectors and triggering probabilities
Older Structured Representations for Activity Recognition

- *Parametric HMMs for “structured” gesture recognition*
  - *Capture the systematic variation of parameterized gesture for *recognition*.*

- Grammar-based representation and parsing
  - Highly expressive for activity description
  - Easy to build higher level activity from reused low level vocabulary.

- P-Net (Propagation nets) – really stochastic Petri nets
  - Specify the structure – with some annotation can learn detectors and triggering probabilities
Anatomy of Hidden Markov Models

- Typically thought of as a stochastic FSMs where:
  - \( a_{ij} := P(q_t = j|q_{t-1} = i) \)
  - \( b_j(x) := p(x_t = x| q_t = j) \)

- HMMs model activity by presuming activity is a first order Markov process and sequence is output from the \( b_j(x) \). States are hidden and unknown.
- Train via expectation/maximization. (EM)
- Paradigm:
  - Training: examples from each class, slow but OK.
  - Testing: fast (Viterbi), typical PR types of issues.
  - Backward looking, real-time at completion.
Wins and Losses of HMMs in Gesture for Recognition (surely not generation?!?!)  

• Good points about HMMs:  
  • A learning paradigm that acquires spatial and temporal models and does some amount of feature selection.  
  • Recognition is fast; training is not so fast but not too bad.  
  • “Generative models” – can always add new classes  

• Not so good points:  
  • If you know something about state definitions, difficult to incorporate (coming later...)  
  • Every gesture is a new class, independent of anything else you’ve learned.  
  • ->Particularly bad for “parameterized gesture.”
Parameterized Gesture

“I caught a fish this big.”
Parametric HMMs

• Basic ideas:
  • Make output probabilities of the state be a function of the parameter of interest $b_j(x)$, becomes $b'_j(x, \theta)$.
  • Maintain same temporal properties, $a_{ij}$ unchanged.
  • Train with known parameter values to solve for dependencies of $b'$ on $\theta$.
  • During testing, use EM to find $\theta$ that gives the highest probability. That probability is confidence in recognition; best $\theta$ is the parameter.

• Issues:
  • How to represent dependence on $\theta$?
  • How to train given $\theta$?
  • How to test for $\theta$?
  • What are the limitations on dependence on $\theta$?
Parametric HMMs

- Basic ideas:
  - Make output probabilities of the state be a function of the parameter of interest $b_j(x)$, becomes $b'_j(x, \theta)$.
  - Maintain same temporal properties, $a_{ij}$ unchanged.
  - Train with known parameter values to solve for dependencies of $b'$ on $\theta$.
  - During testing, use EM to find $\theta$ that gives the highest probability. That probability is confidence in recognition; best $\theta$ is the parameter.

- Issues:
  - How to represent dependence on $\theta$?
  - How to train given $\theta$?
  - How to test for $\theta$?
  - What are the limitations on dependence on $\theta$?

- Basic Idea: Make the HMM’s parameters sensitive the parameter you want to estimate.
Linear PHMM - Representation

Represent dependence on $\theta$ as linear movement of the mean of the Gaussians of the states:

\[
\hat{\mu}_j(\theta) = W_j \theta + \mu_j
\]

\[
P(x_t|q_t = j, \theta) = \mathcal{N}(x_t, \hat{\mu}_j(\theta), \Sigma_j)
\]

Need to learn $W_j$ and $\mu_j$ for each state $j$. 
Linear PHMM - training

• Need to derive EM equations for linear parameters and proceed as normal:

\[ \hat{\mu}_j(\theta) = W_j \theta + \bar{\mu}_j \]  \hspace{1cm} (1)

\[ P(x_t \mid q_t = j, \theta) = \mathcal{N}(x_t, \hat{\mu}_j(\theta), \Sigma_j) \]  \hspace{1cm} (2)

\[ Z_j \equiv \begin{bmatrix} W_j & \hat{\mu}_j \end{bmatrix} \quad \Omega_k \equiv \begin{bmatrix} \theta_k \\ 1 \end{bmatrix} \]  \hspace{1cm} (3)

\[
\frac{\partial Q}{\partial Z_j} = \sum_k \sum_t \gamma_{ktj}(x_{kt} - \hat{\mu}_j(\theta_k))^T \Sigma_j^{-1} \frac{\partial \hat{\mu}_j(\theta_k)}{\partial Z_j} \\
= \Sigma_j^{-1} \sum_k \sum_t \gamma_{ktj}(x_{kt} - \hat{\mu}_j(\theta_k)) \Omega_k^T \\
= \Sigma_j^{-1} \left[ \sum_{k,t} \gamma_{ktj} x_{kt} \Omega_k^T - \sum_{k,t} \gamma_{ktj} Z_j \Omega_k \Omega_k^T \right] \hspace{1cm} (4)
\]

Setting this derivative to zero and solving for \( Z_j \), we get the update equation for \( Z_j \):

\[
Z_j = \left[ \sum_{k,t} \gamma_{ktj} x_{kt} \Omega_k^T \right] \left[ \sum_{k,t} \gamma_{ktj} \Omega_k \Omega_k^T \right]^{-1} \hspace{1cm} (7)
\]
Linear HMM - testing

• Derive EM equations with respect to $\theta$:

$$\frac{\partial Q}{\partial \theta} = \sum_i \sum_i \gamma_{ij} (x - \mu_j(\theta))^T \Sigma_j^{-1} \frac{\partial \mu_j(\theta)}{\partial \theta}$$

$$\theta = \left[ \sum_{t,j} \gamma_{tj} W_j^T \Sigma_j^{-1} W_j \right]^{-1} \left[ \sum_{t,j} \gamma_{tj} W_j^T \Sigma_j^{-1} (x - \mu_j) \right]$$

• We are testing by EM! (i.e. iterative):
  • Solve for $\gamma_{tk}$ given guess for $\theta$
  • Solve for $\theta$ given guess for $\gamma_{tk}$
How big was the fish?

Stereo blob-tracking vision system

Fish size recovery
How does this relate to learning actions for robotics?

- HMMs were never designed to encode the fine grain dynamics. Assumption was that gross trajectory was what mattered.

- Over last several years HMMs have been used to model the trajectories in feature space to be generated. Some additional constraint added to make trajectories have plausible dynamics.

- PHMMs or other variations allow conditioning the trajectories on task variables.

- *But I thought these are quite global effects – not really sufficiently task specific to control execution.*
Statistical dynamical systems for skills acquisition in humanoids

Sylvain Calinon, Zhibin Li, Tohid Alizadeh, Nikos G. Tsagarakis and Darwin G. Caldwell

Abstract—Learning by imitation in humanoids is challenging due to the unpredictable environments these robots have to face during reproduction. Two sets of tools are relevant for this purpose: 1) probabilistic machine learning methods that can extract and exploit the regularities and important features of the task; and 2) dynamical systems that can cope with perturbation in real-time without having to replan the whole movement. We present a learning by imitation approach combining the two benefits. It is based on a superposition of virtual spring-damper systems to drive a humanoid robot movement. The method relies on a statistical description of the springs attractor points acting in different candidate frames of reference. It extends dynamic movement primitives models by formulating the dynamical systems parameters estimation problem as a Gaussian mixture regression problem with probabilistic constraints.

<table>
<thead>
<tr>
<th>Title</th>
<th>Force-based Robot Learning of Pouring Skills using Parametric Hidden Markov Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authors</td>
<td>Rozo, L., Jimenez, P. and Torras, C.</td>
</tr>
<tr>
<td>Year of publication</td>
<td>2013</td>
</tr>
<tr>
<td>Place of publication</td>
<td>IEEE-RAS Intl Workshop on Robot Motion and Control (RoMoCo).</td>
</tr>
</tbody>
</table>

Download the publication in a PDF format
Go to the publisher’s website
Go back to the list of publications
Older Structured Representations for Activity Recognition

- Parametric HMMs for “structured” gesture recognition
  - Capture the systematic variation of parameterized gesture for *recognition*.

- Grammar-based representation and parsing
  - Highly expressive for activity description
  - Easy to build higher level activity from reused low level vocabulary.

- P-Net (Propagation nets) – really stochastic Petri nets
  - Specify the structure – with some annotation can learn detectors and triggering probabilities
Older Structured Representations for Activity Recognition

- Parametric HMMs for “structured” gesture recognition
  - Capture the systematic variation of parameterized gesture for *recognition*.

- Grammar-based representation and parsing
  - Highly expressive for activity description
  - Easy to build higher level activity from reused low level vocabulary.

- P-Net (Propagation nets) – really stochastic Petri nets
  - Specify the structure – with some annotation can learn detectors and triggering probabilities
Semantic (or Task based?) Activities: Known structure, uncertain elements

- Many activities are comprised of a priori defined sequences of primitive elements.
  - Assembly tasks
  - Robot LBD: Washing dishes, setting the table, flipping pancakes
  - The states are not hidden.
- The activities can be described by a set of grammar-like rules: partially ordered, optional steps
- Three approaches:
  - Stochastic Context Free Grammars
  - P-Nets
  - (new!) Sequential Interval Networks
- All handle uncertainty in observation and ambiguity in task
P-Nets (Propagation Networks) (Shi and Bobick, ’04 and ’06)

• Structure known *a priori*.
• Nodes represent *activation intervals* – token propagation.
  • *More than one node can be active at a time!*
• Links represent partial order as well logical constraint
• *Duration model* on each link and node:
  • Explicit model on length of activation
  • Explicit model on length between successive intervals
• Observation model on each node
Propagation Net – Computing

- Computational method
  - A DBN style rollout to compute corresponding conceptual schema. Used a particle filter approach where the state of the particle is the history of the intervals until now.
Example: Glucose Project

- Task: monitor an user to calibrate a glucose meter and point out operating error as feedback.
- Constructed 16 node P-Net as representation

- 3 subjects with total of:
  21 perfect sequences – 5 used for training the detectors.
  10 missing only one step;
  10 missing 6 steps.
Experiment: Glucose Meter Calibration
# Experiment: Classification Performance

<table>
<thead>
<tr>
<th>Labeled Actual</th>
<th>Total</th>
<th>Perfect</th>
<th>Skips, parsed</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>6</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>Perfect</td>
<td>15</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>Missing One</td>
<td>10</td>
<td>20%</td>
<td>80%</td>
<td>%</td>
</tr>
<tr>
<td>Missing six</td>
<td>10</td>
<td>0%</td>
<td>50%</td>
<td>50%</td>
</tr>
</tbody>
</table>
Lesson from PHMMS, SCFGs, and P Nets:

It’s all about the representation ...

• Encoding knowledge of the structure of the activity allows overcoming perceptual uncertainty

• Making the reasoning probabilistic allows handling noise and perceptual error
Now some new stuff...
Anticipating the actions of humans

- Goal: Anticipate the actions of humans such that a robot can anticipate the needs of the human to provide assistance when needed – no waiting.

A challenge because:
- Human collaborator doesn’t do same thing every time, even in assembly situations
- The rate at which they do it varies
- Perception is (usually) an uncertain business
- Robots take time to do things (way too much time)
Integrate over task and perceptual uncertainty

Our method:

1. Compile **structured representations** of activity into **probabilistic** system for reasoning about task and timing
   - *The variables of interest: the stop and end time of sub-actions*

2. Learn from (very small amounts of) data:
   - *Duration models of sub-actions*
   - *Likelihood of branches in activity*
   - *Perceptual detectors* that encode (noisy) information about the human performance (or start and end) of actions

3. At **every time step**, perform **inference** on all actions.

4. Make **plans** based upon probabilistic assessment of what actions will be done and when
Integrate over task and perceptual uncertainty

Our method:

1. Compile **structured representations** of activity into **probabilistic** system for reasoning about task and timing
   - *The variables of interest: the stop and end time of sub-actions*

2. Learn from (very small amounts of) data:
   - *Duration models of sub-actions*
   - *Likelihood of branches in activity*
   - *Perceptual detectors that encode (noisy) information about the human performance (or start and end) of actions*

3. At **every time step**, perform **inference** on **all** actions.

4. Make **plans** based upon probabilistic assessment of what actions will be done and when.
Integrate over task and perceptual uncertainty

Our method:

1. Compile **structured representations** of activity into **probabilistic** system for reasoning about task and timing
   - *The variables of interest: the stop and end time of sub-actions*

2. Learn from (very small amounts of) data:
   - *Duration models* of sub-actions
   - *Likelihood of branches in activity*
   - *Perceptual detectors* that encode (noisy) information about the human performance (or start and end) of actions

3. At **every time step**, perform **inference** on **all** actions.

4. Make **plans** based upon probabilistic assessment of what actions will be done and when
Integrate over task and perceptual uncertainty

Our method:

1. Compile **structured representations** of activity into **probabilistic** system for reasoning about task and timing  
   - *The variables of interest: the stop and end time of sub-actions*
2. Learn from (very small amounts of) data:  
   - *Duration models* of sub-actions  
   - Likelihood of branches in activity  
   - *Perceptual detectors* that encode (noisy) information about the human performance (or start and end) of actions
3. At **every time step**, perform **inference** on **all** actions.
4. Make **plans** based upon probabilistic assessment of what actions will be done and when
Sequential model \((\text{Humanoids 2013})\)

- Discrete time temporal model for task
- Duration model: \(P(g_k^e | g_k^s) \propto D_k (g_k^e - g_k^s)\)
- Sensor model: \(P(Z^k | g_k^s, g_k^e)\) \((\text{non-informative future})\)
- Inference model (basic chain):
  \[
P(g, Z) = \prod_{k=1}^{K} P(g_k^s | g_{k-1}^e) P(g_k^e | g_k^s) P(Z^k | g_k^s, g_k^e)
\]
- At every moment in time can infer the distribution of all the sub-tasks start and end times
Our domain
Illustration of inference
Recall of some of our computer vision work...

- **Parametric HMMs** for “structured” gesture recognition
  - Coupled parametric modeling with *graphical model inference*

- **Stochastic Context Free Grammar** based representation and parsing
  - *Richly expressive for activity description*
  - Easy to build higher level activity from reused low level vocabulary.

- **P-Net (Propagation nets)**
  - Focused on *intervals*
  - Specify the structure – with some annotation can *learn detectors and triggering probabilities*
Grammars: More interesting task descriptions

- First do “a” and “b” in any order, then do “c” and optionally then do “d”
- Can be written as a (trivial) grammar:
  \[ S \rightarrow (a \ b \mid b \ a) \ c \ (d \mid \emptyset) \]
- An AND-OR tree that expresses temporal ordering and selection
From AND-OR to Bayes Networks (ICRA '2014)

**Primitive action v**

**AND: A -> MN**

**OR: A -> M | N**
Some gratuitous math...

The input include: (1) $P(\exists S)$: the prior probability of S happening, (2) $P(S_s|\exists S)$: The prior probability of the start of S, (3) $P(Z_{end}|S_e, \exists S)$: the likelihood representing the constraint on the end of S, and (4) the CPT $P(A_e|A_s)$ and $P(Z^A|A_s, A_e)$ for all primitive action A (recall that our random variables have discrete values between 1 and T. The special value $A_s = A_e = -1$ means $\exists A$, the case where the action A happens)

Step 1, Forward phase: Given $P(A_s, Z^{pre(A)}|\exists A)$, one can compute $P(A_e, Z^{pre(A)}|\exists A)$ for every action A (where $Z^{pre(A)}$ stands for the observation of all actions happening before A). If A is a primitive action, then compute the joint $P(A_s, A_e, Z^{pre(A)}|\exists A)$ and perform marginalization. If A is defined as M AND N, then recursively compute $P(M_e, Z^{pre(A)}, M|\exists M)$ and $P(N_e, Z^{pre(A)}, M, N|\exists N)$ then we have the distribution of $A_e$ the same as $N_e$. On the other hand if A is defined as M OR N, then the distribution of $A_e$ will be weighted combination of $M_e$ and $N_e$ according to equation 3.

The forward process starts with $P(S_s|\exists S)$ and recursively compute $P(A_s, Z^{pre(A)}|\exists A)$, $P(A_e, Z^{pre(A)}|\exists A)$ for every action A.

Step 2, Backward phase: similarly, this process starts with $P(Z_{end}|S_e, \exists S)$ and recursively compute $P(Z^{post(A)}|A_e, \exists A)$, $P(Z^{post(A)}|A_s, \exists A)$ for every action A (here $Z^{post(A)}$ stands for observation of all actions happening after A).

Step 3, compute the posteriors: this is done simply by multiplying the forward and backward messages, we obtain $P(A_s, Z|\exists A)$ and $P(A_e, Z|\exists A)$ for every action A. Additionally we can have $P(Z) = \sum_{t>0} P(S_s = t, Z)$

Step 4, compute the posterior probabilities of an action happening: starting with $P(\exists S|Z) = P(\exists S) = 1$, evaluate $P(\exists A|Z)$ for every symbol A recursively.

Given S is defined as A AND B, then $P(\exists A|Z) = P(\exists B|Z) = P(\exists S|Z)$.

Given S is defined as A OR B, one can compute (apply similar formulas for B):

$$P(\exists A|Z) = P(\exists S|Z) \frac{P(\exists A, Z|\exists S)}{P(\exists A, Z|\exists S) + P(\exists B, Z|\exists S)} \quad (1)$$

where $P(\exists A, Z|\exists S)$ can be calculated:

$$P(\exists A, Z|\exists S) = P(\exists A|\exists S) \sum_{t>0} P(A_e = t, Z|\exists A) \quad (2)$$

Output: the probability of action A happening $P(\exists A|Z)$, and if that the case, the distribution of the start and end $P(A_s, Z|\exists A)$, $P(A_e, Z|\exists A)$. We can compute:

$$P(A_s|Z) = P(\exists A|Z) \frac{P(A_s, Z|\exists A)}{\sum_{t>0} P(A_s = t, Z|\exists A)} \quad (3)$$

for values between 1 and T. Note that $P(A_s = -1|Z) = P(!A|Z) = 1 - P(\exists A|Z)$. 
Evolving prediction uncertainty (CVPR'14): Sequential \textit{Interval} Networks
Parsing (only) video (and more CVPR 2014)
And now apply this to robotics...
Given timing distributions, we need a plan

• Two sources of “cost”: remove a bin early, deliver a bin late. Can be condensed to function of individual wait times:

\[ C = \sum_i \Psi(w_i) \]

where \( i \) is for each time the human needs to wait, \( w_i \) is the amount of wait time \( i \), and \( \Psi \) is sum HRI determined function we used quadratic) Note this is not necessarily total execution time.

• Planning is a heuristic over the independently considered intervals.
Planning in action
Certainty of belief affects plan
Working with the robot
Task indeterminacy
Conclusions

- Representations of activity can make the problem hard or easy. Perhaps *learning paradigm* is trying to automatically extract the representational primitives that make it easy (low dimensions).

- Fundamental is the question of the components of the activity – beyond a simple label. And don’t forget uncertainty (both perceptual and procedural).

- My recent interest: Need to be able to *predict what and when* sub-tasks (or milestones) will occur. The earlier the better. Maybe that’s a deep understanding of action.

- Never too late to become a roboticist.
Done here...