Robot Programming by Demonstration
with Crowdsourced Action Fixes

Maxwell Forbes  Michael Chung  Maya Cakmak  Rajesh P. N. Rao

UNIVERSITY OF WASHINGTON

Human Computation & Crowdsourcing 2014, Pittsburgh, PA
general-purpose robots that can be programmed by their end-users, in the context of use
WHY?

unbounded use cases
WHY?

unique preferences and needs

H. Evans
mute, quadriplegic

O. Benjelloun
BMCE bank chairman
WHY?

simpler engineering challenge

robot that can do everything everywhere - VS - robot that can do a few things in one context
PROGRAMMING BY DEMONSTRATION

Cakmak & Takayama, HRI 2014
**ACTION REPRESENTATION**

\[ \mathcal{A}_n = \{(\theta^f, f, g)_k : k = 1..K\} \]

- **End-effector pose (6-DOF)** gripper state reference frame of landmark \( f \)
- **gripper state index** of the reference frame of landmark
- **Origin**
  - **Landmark pose** (count) descriptor relative to robot origin
End-user programming (EUP) is an active research area for software developers to create custom programs that meet their requirements. The user manipulates a robot, recording their initial configuration. Then the user manipulates the robot's arm, changing its configuration by searching for landmarks in the environment.

The frame of reference for the robot is the landmark descriptors; the type of the landmark and is determined by the robot's current arm configuration.

The user can directly edit components of this representation. These visualizations should be interactive, allowing the user to change the reference frame, remove action states, and delete landmarks.

The action representation described in Sec. II-A. Furthermore, the user can remove landmarks (except robot base). The user can directly edit the configuration of saved action states; transformation of configuration:

\[ d \equiv L_i \cdot \theta \]

where \( d \) is the transformation, \( L_i \) is the type of the landmark, and \( \theta \) is the origin of the robot.

For instance, in spreadsheets the user can program a type-specific, real-valued similarity function. Our work aims to apply the same idea to the EUP literature, allowing the user to edit the robot's representation of the world.

However, more complex functions require the user to edit the visualizations. For example, in spreadsheets the user can program a type-specific, real-valued similarity function to determine the similarity between landmarks of the same type.

The user can remove action states that were created. The user can also directly edit components of the action representation.

The user can also remove landmarks from the set of available landmarks. The similarity between landmarks of the same type is determined empirically.
B. Action Initialization by Demonstration

Actions are initialized with a single demonstration. This process involves directly adding steps into the action representation described in Sec. II-A. Furthermore, the user can directly interact with the robot to edit the action, allowing for the addition, deletion, or modification of landmarks. Once all landmarks are registered, the absolute end-effector configuration into the robot's coordinate frame, \( s_f \), is determined by the transformation function \( T \) for the state, specified as \( \theta \).

C. Action Editing

The robot's relative configuration of the landmark in the environment is determined by the type-specific, real-valued similarity function \( \alpha \). The robot's relative configuration \( v \) is the origin of the action states \( \{ f \} \), where \( f \) is the transformation function. We represent landmarks in the environment as a list of potential landmarks (including the robot base). The robot accumulates the list of potential landmarks by searching for landmarks in the environment using inverse kinematics (IK). If there is at least one pose that reinitializes the action, the action is then updated. Otherwise, the frame of reference is the nearest available landmark (including the robot base). We represent landmarks in the environment as a list of potential landmarks (including the robot base).

D. Action Execution

To perform an action, the user needs to perform a rigid registration between landmarks. Once all landmarks are registered, the absolute end-effector configuration into the robot's coordinate frame, \( s_f \), is determined by the transformation function \( T \) for the state, specified as \( \theta \).

The user can directly interact with the robot to edit the action, allowing for the addition, deletion, or modification of landmarks. Once all landmarks are registered, the absolute end-effector configuration into the robot's coordinate frame, \( s_f \), is determined by the transformation function \( T \) for the state, specified as \( \theta \).

We represent landmarks in the environment as a list of potential landmarks (including the robot base). The robot accumulates the list of potential landmarks by searching for landmarks in the environment using inverse kinematics (IK). If there is at least one pose that reinitializes the action, the action is then updated. Otherwise, the frame of reference is the nearest available landmark (including the robot base). We represent landmarks in the environment as a list of potential landmarks (including the robot base).
We represent landmarks in the environment as a Freedom (DoF) end-effector configuration (translation and rotation) in the frame of reference. If no landmarks are available at execution time, this is done through a simple bi-directional greedy registration between landmarks of the same type in the EUP literature, including programming spreadsheets and webpages. One of the core concerns in EUP is to provide visualizations of robot PbD. To this end, it is important for the system to allow the user to edit the robot's representation of the world: the pose is considered to be referenced in the nearest available landmarks (including the robot base). The user can perform the following edits on the robot's representation:• The user can remove action states that were demonstrated by the user;• The user can remove landmarks (except robot base) that are considered part of the learned action;• During the initial demonstration the user can change the reference frame based on its proximity to landmarks. After the demonstration, the frame of reference is the nearest available landmarks (including the robot base). The algorithm transforms the configuration of the landmark in the robot's coordinate frame, where

\[ \mathbf{T}_i = \mathbf{T}_{0i} \mathbf{T}_{0f} \mathbf{T}_{f} \mathbf{T}_{g} \]

\[ \mathbf{T}_{0i} \]

is the transformation of configuration: Transformation of configuration:

\[ \mathbf{T}_i = \mathbf{T}_{0i} \mathbf{T}_{0f} \mathbf{T}_{f} \mathbf{T}_{g} \]

\[ \mathbf{T}_{0i} \]

is computed by transforming the absolute end-effector configuration into the action states.

\[ \mathbf{T}_i \]

represents its binary gripper state (open or closed). We represent landmarks in the environment as a Freedom (DoF) end-effector configuration (translation and rotation) in the frame of reference, where

\[ \mathbf{T}_i = \mathbf{T}_{0i} \mathbf{T}_{0f} \mathbf{T}_{f} \mathbf{T}_{g} \]

\[ \mathbf{T}_{0i} \]

is computed by transforming the absolute end-effector configuration into the action states.

\[ \mathbf{T}_i \]

represents its binary gripper state (open or closed). We represent landmarks in the environment as a Freedom (DoF) end-effector configuration (translation and rotation) in the frame of reference.
ACTION EXECUTION

\[
\begin{align*}
\theta_0^0 & \\
\ell_0 & \text{ (origin)} \end{align*}
\]

\[
\begin{align*}
\theta_1^1 & \\
\ell_1' & \\
\theta_2^1 & \\
\ell_2' & \\
\theta_3 & \\
\ell_3' & \\
\end{align*}
\]
SIMPLE TASK
MORE DIVERSE TASKS
PROBLEM: GENERALIZATION
END-user programming (EUP) is an active research area that aims to enable everyday people who are not professional software developers to create custom programs that meet their needs. We represent landmarks in the environment as a matrix corresponding to the configuration \( f \) of the landmarks in the current environment. Otherwise, the robot reproduces the action as previously demonstrated.

In addition to these edits on the programmed action, we allow the user to perform the following edits on the desired arm pose:

• **Delete landmark:** Removes an existing landmark.
• **Delete pose:** Removes a pose from the action.
• **Reference frame change:** Changes the reference frame of the action.
• **Edit:** Directly edits components of the representation.

A demonstration consists of a sequence of robot poses, labeled \( \{L_n\}_{n=0}^N \), which is a type of function \( \mathbb{R}^N \to \mathbb{R}^N \). Actions are initialized with a single demonstration. This registration is determined by the robot's current arm configuration \( \theta \). After an action has been initialized, it can be cleared and re-initialized, edited (Sec. II-C), or executed (Sec. II-D). The similarity \( s_{d,n} \) of a pose \( d \) to a pose \( n \) is computed by transforming \( d \) into the absolute end-effector configuration \( f_{d} \) and computing the robot's 6 Degree-of-Freedom (DoF) end-effector configuration (translation and orientation).

Actions are initialized with a single demonstration. This representation of the action consists of a set of landmarks \( L_0, \ldots, L_N \) and a set of angles \( \theta_0, \ldots, \theta_N \). The similarity \( s_{d,n} \) of a pose \( d \) to a pose \( n \) is computed by transforming \( d \) into the absolute end-effector configuration \( f_{d} \) and computing the similarity between landmarks of same type in \( L_0, \ldots, L_N \). The user can remove action states that were registered. If at least one landmark in \( \{L_n\}_{n=0}^N \) is considered to be another available landmark, the system greedily removes the pair of landmarks with highest similarity.

Implementation of our approach to provide visualizations of the user with an accurate mental model of what the system aims to do is determined by the robot's current arm configuration. At any time, the current action state is the last recorded steps to robot PbD. To this end, it is important for the system to allow the user to edit the robot's representation of the world.
PROBLEM: GENERALIZATION
PROBLEM: GENERALIZATION

\[ \begin{align*}
\theta_0^0, \theta_1^1, \theta_2^1, \theta_3^2, L_1', L_2', L_0 (\text{origin})
\end{align*} \]
PROBLEM: GENERALIZATION

The frame of reference is determined by the robot's current arm configuration. The pose is considered to be relative to robot PbD. To this end, it is important for the system to robot with an accurate mental model of what the system can represent so as to allow them to directly edit components of this representation.

During the initial demonstration the user can change the reference frame greedily removes the pair of landmarks with highest similarity between landmarks of same type in the current configuration of the landmarks. The desired arm action by demonstrating each state. Before a demonstration, the user manipulates the robot's arms and starts to add steps into the action configuration (translation and rotation) in the frame of reference.

End-user programming (EUP) is an active research area that aims to enable everyday people who are not professional software developers to create custom programs that meet their needs. One of the core concerns in EUP is to provide simple functions by providing example input-output pairs; these visualizations should be interactive, to allow the user to record the initial configuration. Then the user manipulates the robot's arms and starts to add steps into the action configuration (translation and rotation) in the frame of reference.

The user can remove action states that were re-initialized, edited (Sec. II-C), or executed (Sec. II-D).

The type of the landmark and its absolute descriptor of the landmark consisting of a vector of reals.

• Delete pose: During the initial demonstration the user can change the reference frame and recording their initial configuration. Then the user manipulates the robot's arms and starts to add steps into the action configuration (translation and rotation) in the frame of reference.

Actions are initialized with a single demonstration. This is the type of the landmark and its absolute descriptor of the landmark consisting of a vector of reals.

...
OBJECTIVES

• Improve range of actions

• Impose no additional requirements on users
APPREACH

1. Start with a single demonstration
2. Instance-based active learning
3. Crowdsourc source additional demonstrations
APPROACH

1. Start with a single demonstration

2. Instance-based active learning

3. Crowdsourse additional demonstrations
the robot's coordinate frame, i.e., the pose is considered to be at the proximity of the end-effector to the landmarks. If no landmarks are available, the frame of reference is the nearest.

The frame of reference is the nearest.

Actions are initialized with a single demonstration. This involves the user manipulating the robot while recording their initial configuration. Then the user edits the configuration of saved action states.

During the initial demonstration, the robot accumulates the list of potential landmarks.

Once all landmarks are registered, the absolute end-effector configuration to achieve each end-effector pose is computed using inverse kinematics (IK). If there is at least one pose that can be achieved, the action is deemed executable in the current environment. If at least one landmark in the environment is a type-specific, real-valued similarity function between two landmarks of the same type is computed with empirical determination.

During the execution of actions, these visualizations should be interactive, to allow the user to directly edit the configuration of saved action states. The user can remove action states that were unnecessary or incorrect.
reals. or closed). We represent landmarks in the environment as a Freedom (DoF) end-effector configuration (translation and rotation) in the frame of reference $\mathbb{SE}(3)$, where

$$
\begin{align*}
\theta_i &= \text{rotation} \\
T_i &= \text{translation}
\end{align*}
$$

End-user programming (EUP) is an active research area for software developers to create custom programs that meet their particular needs \[7, 17, 9\]. Commonly studied problems in the EUP literature include programming spreadsheets and webpages. One of the core concerns in EUP is to provide visualizations of these visualizations should be interactive, to allow the user to edit the robot's representation of the world:

- **Delete landmark:** Delete an available landmark to be another available landmark;
- **Delete pose:** Delete a specific pose to be another available landmark;
- **Delete action:** Delete an entire action to be another available action;
- **Replace landmark:** Replace a landmark with another landmark;
- **Replace pose:** Replace a pose with another pose;
- **Replace action:** Replace an entire action with another action;
- **Add landmark:** Add a new landmark to the list of potential landmarks;
- **Add pose:** Add a new pose to the list of potential end-effector configurations;
- **Add action:** Add a new action to the list of potential actions;
- **Clear action:** Clear the list of potential actions.

After an action has been initialized, it can be cleared and saved during the demonstration; the EUP literature includes programming spreadsheets and webpages. Our work aims to apply the same idea and allow the user to directly edit representations so as to allow them to directly edit representations for functions. During the initial demonstration the user can change the reference frame based on its proximity to landmarks. After the demonstration the robot first accumulates the list of potential landmarks, i.e., $\mathbb{L}_d$, which contains all landmarks that are considered part of the learned action:

$$
\mathbb{L}_d = \text{arg min}_{\mathbb{L}} \sum_{i=1}^{n} d_i
$$

where $d_i$ is the type of the landmark and $f_i$ is the end-effector configuration of the landmark in the robot's coordinate frame, i.e., the pose is considered to be $\mathbb{L}_d$. Otherwise, the frame of reference is the nearest base configuration of the landmark in the robot's coordinate frame, i.e., the pose is considered to be $\mathbb{L}_d$. The similarity between two landmarks of the same type is computed with the formula of a function.
IMPLEMENTATION

1. Collect single demonstration of three actions
IMPLEMENTATION

1. Collect single demonstration of three actions
IMPLEMENTATION

1. Collect single demonstration of three actions

2. Filtered random sampling for active learning
left arm range

right arm range

robot facing table
left arm range  

robot facing table  

right arm range
left arm range

right arm range

0 unreachable

robot facing table
FILTERED RANDOM SAMPLING
IMPLEMENTATION

1. Collect single demonstration of three actions

2. Filtered random sampling for active learning
IMPLEMENTATION

1. Collect single demonstration of three actions

2. Filtered random sampling for active learning

3. Collect and process data
   - GUI
GUI

If time, make slide about color coding of grippers.
GUI

Step 1

Step 2

Step 3

Step 4

Step 5
IMPLEMENTATION

1. Collect single demonstration of three actions

2. Filtered random sampling for active learning

3. Collect and process data
   • GUI
IMPLEMENTATION

1. Collect single demonstration of three actions

2. Filtered random sampling for active learning

3. Collect and process data
   - GUI
   - Score functions
SCORE FUNCTIONS

1. Crowd confidence

\[ s_o(D_i) = conf(D_i) \]
1. Crowd confidence

\[ s_o(D_i) = \text{conf}(D_i) \]

2. Seed distance

\[ s_d(D_i) = \frac{1}{\sum_{j=1}^{N_j} \frac{|\theta_{0,j} - \theta_{i,j}^f|}{|\theta_{0,j}^f|}} \]
SCORE FUNCTIONS

1. Crowd confidence

\[ s_o(D_i) = \text{conf}(D_i) \]

2. Seed distance

\[ s_d(D_i) = \frac{1}{\sum_{j=1}^{N_j} \frac{|\theta_{0,j}^f - \theta_{i,j}^f|}{|\theta_{0,j}^f|}} \]

3. Compactness

\[ s_c(D_i) = \frac{1}{\sum_{j=1}^{K} |\theta_{i,j}^f|, f \neq \ell_0} \]
1. Actions

2. Active learning

3. Crowd data
EVALUATION

Part 1
- Thank you for participating in our study.
- This is Rosie the Robot.
- The goal of our research is to enable everyday people to program new actions on general purpose robots like this one, so that it doesn't need to be preprogrammed for every possible action users might need.
- An intuitive way to program the robot is to just demonstrate the desired action.
- So in our previous research we have had people come in and program new actions by physically moving the robot's arms and using simple commands.
- This is called programming by demonstration.
- The way that it works is: the person saves a sequence of hand poses and then the robot moves its arms to go through those poses.
- Some poses are attached to objects so that the action will work even when the objects move around.
- Here is a video that explains this system.
- Have them watch the PhD intro video (you might want to narrate as you go):
  https://www.youtube.com/watch?v=F0r7rG3ZI0

Part 2
- As you saw in the video, sometimes Rosie is not able to perform the actions that it was programmed to do correctly.
- This can happen when the objects are moved around.
- But as shown in the video you can edit the programmed action through the user interface to make it feasible in that scenario.
- That's exactly what we'll ask you to do today.
- Rosie has already learned three different actions but it automatically found some scenarios in which it is unable to perform each action.
- So you will go in and fix the action to work in those scenarios.
  We have a few examples of this to do.
EVALUATION

Part 1
- Thank you for participating in our study.
- This is Rosie the Robot.
- The goal of our research is to enable everyday people to program new actions on general purpose robots like this one, so that it doesn’t need to be preprogrammed for every possible action users might need.
- An intuitive way to program the robot is to just demonstrate the desired action.
- So in our previous research we have had people come in and program new actions by physically moving the robot’s arms and using simple commands.
- This is called programming by demonstration.
- The way that it works is the person saves a sequence of hand poses and then the robot moves its arms to go through those poses.
- Some poses are attached to objects so that the action will work even when the objects move around.
- Here is a video that explains this system.
  
  Have them watch the PhD intro video (you might want to narrate as you go):  
  https://www.youtube.com/watch?v=Io7o3ex277s

Part 2
- As you saw in the video, sometimes Rosie is not able to perform the actions that it was programmed to do correctly.
- This can happen when the objects are moved around.
- But as shown in the video you can edit the programmed action through the user interface to make it feasible in that scenario.
- That’s exactly what we’ll ask you to do today.
- Rosie has already learned three different actions but it automatically found some scenarios in which it is unable to perform each action.
- So you will go in and fix the action to work in those scenarios.
EVALUATION

Part 1
- Thank you for participating in our study.
- This is Rosie the Robot.
- The goal of our research is to enable everyday people to program new actions on general purpose robots like this one, so that it doesn’t need to be preprogrammed for every possible action users might need.
- An intuitive way to program the robot is to just demonstrate the desired action.
- So in our previous research we have had people come in and program new actions by physically moving the robot’s arms and using simple commands.
- This is called programming by demonstration.
- The way that it works is: the person saves a sequence of hand poses and then the robot moves its arms to go through those poses.
- Some poses are attached to objects so that the action will work even when the objects move around.
- Here is a video that explains this system.
- Have them watch the PhD intro video (you might want to narrate as you go):
  https://www.youtube.com/watch?v=-Ia79grx270

Part 2
- As you saw in the video, sometimes Rosie is not able to perform the actions that it was programmed to do correctly.
- This can happen when the objects are moved around.
- But as shown in the video you can edit the programmed action through the user interface to make it feasible in that scenario.
- That’s exactly what we’ll ask you to do today.
- Rosie has already learned three different actions but it automatically found some scenarios in which it is unable to perform each action.
- So you will go in and fix the action to work in those scenarios.
  https://www.youtube.com/watch?v=3h70t0u0
METRICS

Reachability
(100 scenarios / action)

Success
(10 scenarios / action)
EVALUATION

• Directions, videos, ~45 minutes of work

• 31 people \times 15 \text{ demonstrations each} = 465 \text{ demonstrations in total}

• Metrics
  • reachability \ (100 \text{ scenarios / action})
  • success \ (10 \text{ scenarios / action})
FINDINGS: REACHABILITY

![Graphs showing the reachability over the number of crowd demonstrations for different actions and difficulties.]

- **Action 1**: Demonstrations increase the reachability rapidly, reaching near-perfect scores with a small number of demonstrations.
- **Action 2**: Demonstrations have a moderate impact on reachability, with scores improving steadily over time.
- **Action 3**: Demonstrations show a slower rate of improvement, with the reachability increasing gradually.

Each graph includes a legend indicating the difficulty levels, with 1 being the easiest and 5 the most difficult.
FINDINGS: SUCCESS

The bar chart shows the number of successful out of 10 tests for different score functions across three actions: Action 1, Action 2, and Action 3. The score functions are denoted as $S_o$, $S_d$, and $S_c$. The chart compares confidence, seed distance, compactness, and top 5 performances.
FINDINGS CROWD EFFECTIVENESS

The graph shows the effectiveness of different actions (labeled as Action 1, Action 2, and Action 3) across varying levels of difficulty. The y-axis represents crowd effectiveness, while the x-axis indicates the difficulty levels ranging from 1 to 5. The graph illustrates how effectiveness decreases as difficulty increases for all actions.
FINDINGS: CROWD LEARNING

[Graphs showing average marker fix time for different users across various scenarios.]
FINDINGS: CROWD SCORING

The chart illustrates the confidence levels from 0-100 across varying levels of difficulty. The colors and segmentation of the chart indicate different confidence intervals, with darker shades representing higher confidence levels.
FINDINGS

- Reachability improves with more data
- **Achieved >= 70% on success metric**
- Choice of score function is difficult, important
- Crowd learns
- Need coarser-grained crowd rating system
LIMITATIONS

- No automatic success testing
- Difficult UI
- Crowd data collected in batch mode
- Not real crowdsourcing
SUMMARY

• Goal: **Improve generalizability** of actions in robot programming by demonstration

• Instance-based active learning

• Crowdsourced (~450) demonstrations

• Improved task reachability and success
Thank you

Maxwell Forbes | mbforbes
Michael Chung | mjyc
Maya Cakmak | mcakmak
Rajesh P. N. Rao | rao