Glovebox Handling of High-Consequence Materials with Super Baxter and Gesture-Based Programming - 18598

Teerachart Soratana*, Mythra V. S. M. Balakuntala*, Praveen Abbaraju*, Richard Voyles*, Juan Wachs*, and Mohammad Mahoor**
*Purdue University
** University of Denver

ABSTRACT

The handling of high-consequence materials is a difficult task requiring safe object manipulation while avoiding the risk of contamination or spillage. Specific operations including opening containers, segregating the wastes, and repackaging are required to be safely executed inside contained spaces such as gloveboxes. However, the workers' ability and dexterity to manipulate objects through thick protective gloves are compromised in many aspects. The fixed position of the glovebox's arm ports restricts the movements of the workers' arms, which also makes it hard to lift heavy objects. Further, the operational workspace inside the glovebox is restricted by the gloves' reachability. Safety of workers is the paramount concern in glovebox operations and the very reason for their existence. Sharp edges and tools increases the risk of glove punctures, which may expose the operator to chemicals and radiation and risks contamination in the vicinity outside the glovebox. The operators are also affected by ergonomic stressors due to prolonged and repetitive operations.

To achieve a high degree of human safety, robotic solutions to handle high-consequence materials inside the glovebox are desirable, as they remove the operators from the hazards listed above. However, robots, in general, lack the degree of adaptability necessary for most high-consequence material handling tasks as these tasks are often unstructured or highly variable. Likewise, most human operators, while highly skilled in the tasks at hand (task expertise), lack the robot programming skills necessary to adapt the robot's motions to the variations (programming expertise). A way to bridge this divide is to make robot programming more intuitive to human operators. Since humans naturally teach one another through demonstration and learning, robotic "programming by demonstration" paradigms have begun to appear to reduce the burden of robot reprogramming. *Gesture-Based Programming* attempts to achieve that by enabling a robot to observe the normal actions and affordances of a human performing a task to learn to map those onto the skills of the robot. In the end, this approach permits benefits from both the human and the robot: the human's adaptive decision making and the robot's resilience to high-consequence material.

Robotic solutions assuringly provide protection for the operators, however their autonomy brings about other potential risks. *Artificial Intelligence* is not perfect; it is a black box, which can produce results, but the full extent of those results may not be known precisely. In self-driving cars, a hypothetical AI problem is the "green man detector." Since self-driving cars have presumably never seen a green man, how can we know it won't veer toward him the first time it sees a green man? Likewise, even though GbP provides an open method for safe task execution replicating a human, incorrect task inferences can lead to potential collisions or unintended operations. Thus, a hardware safety measure is proposed to provide reliable operation inside the glovebox, and true safe operations are achieved by mechanically limiting the operational range of each arm link. The operational range is computed and validated through exhaustive offline simulations, which can be streamlined by affordable computing power.

Super Baxter is one such robot which incorporates these capabilities. It is a human-like, bimanual robot, being developed at the Collaborative Robotics Lab (CRL) in conjunction with the Dept. of Energy/Environmental Management, Rethink Robotics, Barrett Technology, and the NSF Center for Robots and Sensors for the Human Well-Being (RoSe-HUB). Super Baxter is envisioned to represent the next generation of collaborative robots that will be intrinsically human-safe, as well as exceptionally human-intuitive.

In summary, Super Baxter will enable safe glovebox operations through low-shot Gesture based Programming incorporated with hardware safety measures. Two aspects of safety will be discussed in the paper. First, the safety of the operator achieved by allowing him to program Super Baxter remotely through Gesture based Programming. Second, the safety of the glovebox, achieved by incorporating hardware safety measures using joint limits. Further, the capability of Super Baxter on Gesture based Programming is demonstrated through typical object manipulation tasks in glovebox operations.

INTRODUCTION

Gloveboxes have become an integral part of laboratory equipment used for safely handling hazardous materials in the nuclear facilities and biological labs. Even though there have been advances in glovebox designs to increase the safety and efficiency, they do not completely eliminate the risk of injuries. The workers are still affected from handling the hazardous substances, ergonomics injuries, etc. There are regulations imposed by the Occupational Safety and Health Administration (OSHA) Standard No. 1926.1101, [1] on handling the toxic and hazardous substances. OSHA regulation 1926.1101(g)(9)(iii) states that if the operation involves processes that could potentially puncture the glove bag such as cutting, drilling, breaking, or sawing, then the operator shall use impermeable protective gear and isolate the operation using a mini-enclosure. There is a risk of injury or exposure in spite of the regulations and safety procedures.

Investigations and surveys have reported injuries and exposure inside gloveboxes to workers at nuclear facilities [2][3]. The reported cases of glovebox related injuries usually resulted from puncture wounds, which lead to internal contamination. Cases of ergonomic injuries resulting from repetitive motion in glovebox operation have been reported. To eliminate risk of injuries to workers while operating the glovebox, it is recommended to evacuate the worker from physical contact with high-consequence material handling.

One solution to high-consequence material handling is robotic teleoperation, which allows the worker to manipulate the objects in the glovebox without exposing him/her to the internal contamination or radiation. There are many examples of using teleoperated robots at nuclear facilities. The workhorse of the nuclear industry has been the CRL Model M2 master-slave manipulator system developed by the Central Research Laboratory and Oak Ridge National Laboratory [4]. A more modern example is the Industrial Reconfigurable Anthropomorphic Dual-arm (IRAD) which is a teleoperated robot designed to manipulate high-consequence material inside the glovebox [5]. IRAD is comprised of two Yaskawa Motoman SIA5 industrial manipulators. It was equipped with the force-torque sensors and 3-fingered adaptive robot gripper. The IRAD system can be teleoperated using three different interfaces: Command line, Graphical User Interface (GUI), and optical hand tracking. GUI was found to be more effective compared to optical hand tracking approach. These teleoperated robots do enhance the safety and efficiency of the workers, but there is huge amount of learning and familiarization required to operate the robot. There are also correspondence issues between the master interface and the slave robot.

A better solution is fully autonomous robots that tirelessly perform tedious functions with precision, under human supervision. Yet, the current state of industrial robots requires highly-structured workcells and highly-experienced programmers to program robots.

For example, workers at nuclear facilities have implicit knowledge of the performed tasks, but lack the necessary syntactic programming skills. Humans are capable of performing complicated tasks like planning and manipulation easily, but describing tasks to the robot by syntactic programming is daunting. On the other hand, humans are very good in teaching by demonstration or with gestures. Gesture based Programming (GbP) is an approach that allows the user to teach the robot by performing the task with gestural language. There are many approaches to facilitate robots programming, an overview of these can be found in [6]. As a platform to implement GbP, we developed the Super Baxter, a bimanual configuration robot with a central head neck system, similar to a human. The reason we integrate the concept of head-neck mechanism displaying emotions is to introduce a socially driven behavior in the robot, which enables the robot to have sustained interaction. Correspondingly, natural communication between the robot and human is essential in teaching process; our goal is to have the human teaches Super Baxter on the same way as they would teach a fellow human co-worker. Thus, interactive communication, which human uses in everyday life, is a natural approach for human co-worker to teach the robot without prior coding experience requirement.

SUPER BAXTER

Super Baxter is designed to be a collaborative robot -- implying it is intrinsically human-safe --- that is aware of human actions and human affective state and capable of interacting with humans in a socially-relevant way. This is important, even in industrial settings, because humans are social animals and have been shown to be more efficient learners and performers when engaged in a socially-appropriate manner [7]. As such, Super Baxter is approximately human-size and has a human-like shape: two arms, multi-fingered hands, and a face above the middle of the torso. (See Fig. 1.) This human-like configuration is beneficial because it makes the mapping of actions between human and robot more direct and more symmetric and the bi-manual aspect allows the robot to provide and control its own fixturing. Moreover, this allows humans to intuitively predict the trajectory of Super Baxter, even before it moves, and to guide it to correct the trajectory and improve its task performance.

Each arm of the Super Baxter is a Rethink Robotics' Sawyer^{TM} arm with Barrett Technology's BH-282^{TM} gripper as the end effector. A Rethink Robotics' Sawyer has 7 degrees of freedom (DOF) and maximum reach of 1.26 m. This large range of reach and 7-DOF give Super Baxter large envelope of reachable workspace and ability to freely manipulate the object inside the envelope. The Sawyer Arm has repeatability of ± 0.1 mm, which enable Super Baxter to perform repetitive tasks with sufficiently good accuracy. Sawyer robot is accredited for ISO 10218-1:2011 to meet international safety requirements for industrial robots. This certifies that the robot is safe to perform delicate tasks that require safety to both humans and equipments. Each arm has 5 kg payload, which makes Super Baxter suitable for light but repetitive tasks performed in the glovebox. In order to increase task manipulation performance, we use a tactile sensing end effector on the Barrett Technology's BH^{TM}-282 grippers. Theses hands have three fingers built in with tactile sensor arrays on the fingertips and palm of the hand. The tactile sensor arrays provide pressure feedback, which enabling Super Baxter to perform tasks like manipulating low stiffness objects or confirming contacts. Each hand has 8 DOF, which allows for dexterous manipulation of objects in the workspace. Moreover, Super Baxter is equipped with an external KINECT^{TM} to detect the target objects and motions in the workspace.

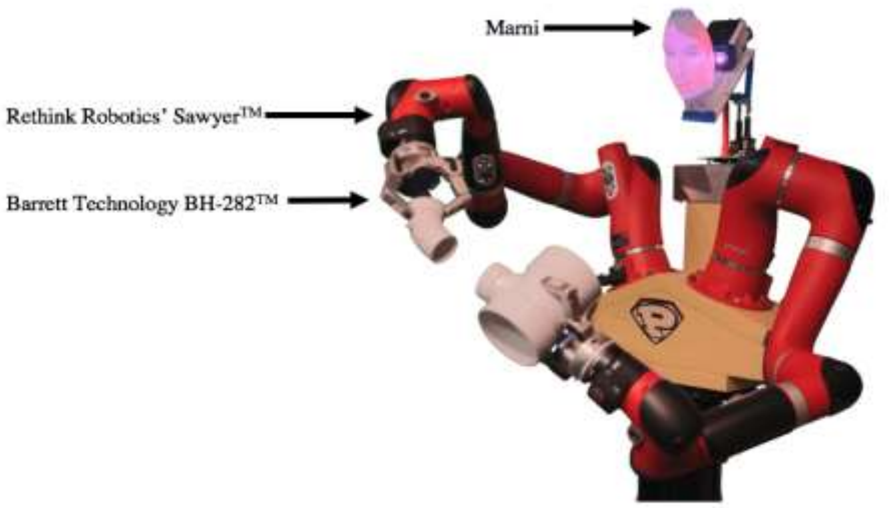

Fig. 1. Super Baxter - a dual arm robot with human-aware capabilities for social interaction

Super Baxter has a human-like face with head-neck mechanism named Marni. Marni is a social avatar projected onto a human-face shaped translucent mask. The mechanism, which supporting the mask, is a 3-DOF platform, allowing Marni to perform neck gestures, augmenting the communication with human co-workers. The head-neck mechanism is controlled using the RecoNode, which is a multi-processor architecture Virtex^{TM} 4 FPGA with multiple hardcore PowerPCs [8]. The face projection on the mask allows Marni to produce human-accustomed facial expressions and eye gazing through rear projection. The social avatar, while aesthetically pleasing, is also more cost-effective [9] and easy to maintain. Super Baxter with Marni as a social interface can use intuitive communication dialog and gestures to communicate with the worker. This potentially reduce the communication load and appears more friendly and approachable to human co-workers.

There is an emerging trend of using social avatar in robots. Rethink Robotics' Baxter and Sawyer are examples of the robots, which use a 2D screen to display the social avatar while operating. Using a screen to display an emotive face increases engagement and awareness [10]. Marni also uses eye gaze direction to improve communication with human co-workers. User studies on 3D physical social avatar show improved human perception and engagement with mutual eye gaze and facial expressions on robot [11]. Natural eye gaze is an important aspect of interactive communication and elicits a more involved conversation [12]. By improving the eye gaze direction perceived by the human co-workers, we can establish more realistic verbal communication with human and make Super Baxter more human-friendly.

GESTURE-BASED PROGRAMMING

The software suite of Super Baxter is what will truly differentiate it as a next-generation collaborative robot. To coexist naturally with humans in a way that boosts the performance of human-robot teams, the advanced capabilities currently under development include human activity recognition [13], facial expression recognition and display [10][14], gesture-recognition for non-verbal communication [15][16][17], natural language processing for spoken dialog [18], and intuitive methods for programming and adaptation [19]. The focus of this section is on the Gesture-Based Programming (GbP) system that provides a programming interface more attuned to ta*sk experts* rather than *programming experts*, allowing operators with minimal training to safely and reliably re-program the robot by showing it what to do.

GbP uses task demonstrations, performed by the operator, to infer the task and map it onto actions the robot already knows. Tasks are assumed to be composed of combinations and sequences of *a priori* skills, which are pre-programmed sensorimotor skills that the robot can perform.

Programming by demonstration paradigms generally use the same environment for both human and robot, but one of the novelties of this work is that the demonstration of the task can be performed in a simulated and inert environment. This allows the human operator to be completely excluded from the glovebox operations.

GbP is more intuitive to human and can be performed with little to nil training. It is not based on record and playback methods, but it is a learning method and thus implies generalization. The robot identifies the task and performs it rather than trying to replicate the human motion recorded by demonstration. GbP enables the robot to derive task inferences and perform the task by observing the human performing it, rather than requiring the end user to break down the task and program the robot. The task is identified as a composition of pre-learnt tasks or *a priori* skills. The aim of GbP is to easily extend the capabilities of robot and make it more adaptable to new situations or tasks. In case the robot is unable to perform the task, the user just needs to provide more demonstrations of the task to improve the task-learning of the robot.

Literature Review

Early works on learning by demonstration or programming by demonstration were approached based on symbolic reasoning with processes like playback methods [20]. The demonstration was broken down into a sequence of state-action-state transitions, which was further converted to if-then rules. These rules described the states and corresponding actions to be performed. In addition, machine learning based approaches have been used as methods for task inference and identification [21]. The approach identified basic operations to define a set of motor skills required for industrial robots. Later approaches incorporating neural mechanisms in animals and children have also been proposed [22], [23]. These approaches lead to robot programming by demonstration being referred to as ‘imitation learning’. These early works determine issues related to programming by demonstration such as generalization of task and reproduction of a completely new task.

Some of these issues can be solved by choosing the right interface for programming by demonstration. Multiple interfaces have been suggested for GbP like sensorized gloves to estimate the pose of the hand [24], vision [25], speech command, and kinesthetic teaching [26] where the user manually holds the robot hand guides it to perform the task. GbP has progressed from simply copying the kinematics to generalizing across several demonstrated tasks. Mapping tasks and motions observed to the robot, which has a different physical structure, has a lot of problems [27]. These problems are broadly referred to as correspondence issues. To overcome correspondence issues, Super Baxter is pre-trained to learn the correspondences between human motion, task kinematics, and its own motion.

Gesture-Based Programming Architecture

In this section we describe the architecture for GbP. the This involves two main steps; human task inference, Task Representation, and robot task planning. Human task inference infer the task based on the demonstration and then robot task planning converts the task to robot executable sequence. This is followed by execution of the task with data from sensors to augment the task execution. Fig. 2 shows the data flow diagram for GbP.

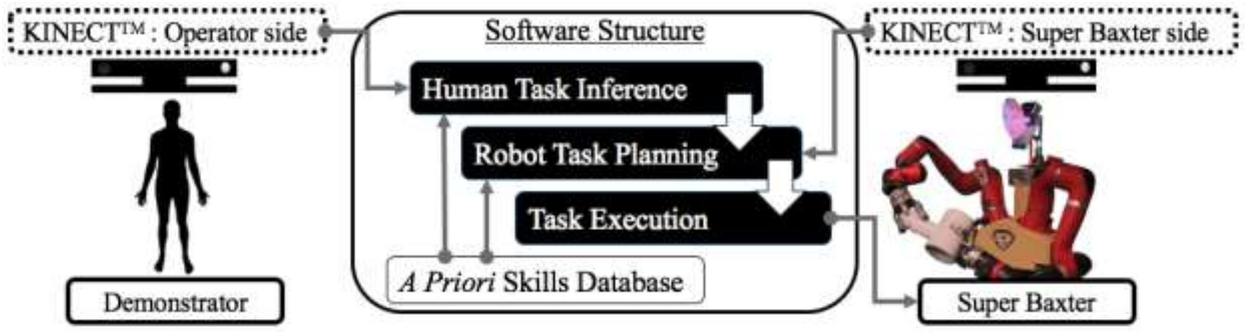

Fig. 2. Software data flow

Human Task Inference and Task Representation
Human task inference is the process of inferring the task demonstrated. The first step is the observation of human skeletal motion, object motion, and task kinematics. We use KINECT^{TM} to get information regarding the human skeleton motion, visual scene, and kinematics of the object using the camera and depth sensors in KINECT^{TM}. This information along with the *a priori* skills is used to infer the task intent and composition.

The task generalization is based on the representation of *a priori* skills. The relative motion between the hand and task objects are used as the basis for the representation. *A priori* skills are represented as a set of variables σ. Where σ is $\{\dot{r}, r, m, s\}$, r denotes the set of relative positions $\{r_1, r_2, \dots . r_n\}$, where r_k is the relative position between the hand and object k. $\dot{r} = \{\dot{r}_1, \dot{r}_2, \dots, \dot{r}_n\}$ is the rate of change of relative position which is relative motion. m is set of additional miscellaneous variables like the start position, end position, relative position between objects, object orientations, etc. The variables in m. s is the previous state of the robot, i.e. the previous *a priori* inferred. Each *a priori* is based on conditions on $\{\dot{r}, r\}$

For example, *a priori* 'transport' of kth object is identified by $\{\dot{r}_k = 0, r_k < \epsilon, m = \{\}\}$. ϵ is a small threshold on position to detect whether the hand and object are very close. If the variable m is null set, it implies the start and end points are not important, which is the case for "transport" task. The variables in m are determined using few repeated demonstrations to arrive at correct inferences. For example, if the objects have to be placed at a specific position, the significance of the position is identified using multiple demonstration of the same task. In this case m will contain an end position variable. If the end position is not important, but the objects have to be moved to an area in general, such as any point on conveyor, then the end point is not important. In this case m will not contain any end position variable.

Another example of *a priori* skill is retract which is identified by $\{\dot{r}_k > 0, r_k > \epsilon, m = \{\}\}$.. For *a priori* skill like stack m will contain the variables for relative positions between objects.

Here we present the task inference for a simple pick and place task. Super Baxter identifies a simple pick and place task as a composition of observable *a priori* skills as shown in Fig. 3. The human skeleton information and object detection from visual sensor can be used to identify the task gestures. These gestures can be recognized as approaching, transporting, releasing the object, or others pre-learnt robot skills (*a priori* skills). The basis for this inference is the representation of the *a priori* skills.

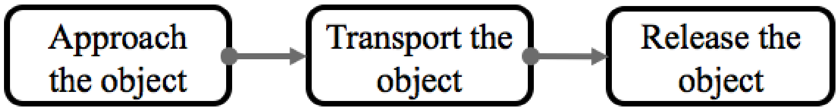

Fig. 3. Observable *a priori* skills from human demonstrator in pick and place task.

Robot Task Planning

Robot Task Planning generates the task models, which is a sequence of *a priori* skills to complete the task. This is different from task inference. Here we use the outputs of task inference to generate a sequence of *a priori* skills to be executed to accomplish the task. The differences are demonstrated using example of the pick and place task in Fig. 4. Some of the *a priori* skills have required conditions to be fulfilled before executing, such as "approach the object" requires the position of the targeted object, and "transport the object" requires the hand to be pre-shaped first and then a grasp to be performed to pick up the object. Thus robot task planning is generating this complete task model, by incorporating the 'transition *a priori* skills'.

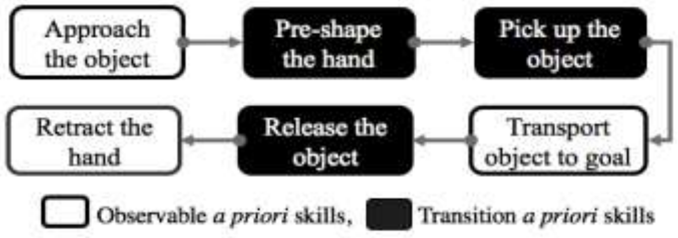

Fig. 4. Pick and place task execution sequence with transition *a priori* skills.

To show the scalability of the task inference and planning, an example of scooping / unscooping task model is shown in Fig. 5. This task is comparatively more complex than pick and place task as it has more steps to complete the task and requires tool manipulation.

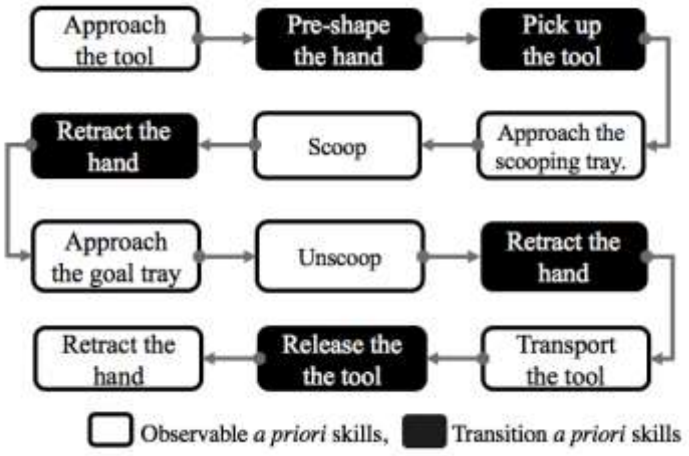

Fig. 5. Scooping / unscooping task execution sequence.

A Priori Skills Database

The *a priori* skills are defined as the basic robot skills or capabilities. These *a priori* skills are used to assemble the 'task model,' which is the representation for a particular task. The *a priori* skills database is the set of these pre-learnt *a priori* skills. The skill's database is also used for developing the task execution sequence, defined as the set of state-action transitions the robot has to perform to execute the task.

The *a priori* skills are separated into two sets, the observable *a priori* skills and the transition *a priori* skills. The observable *a priori* skills are those that can be observed/inferred by using the data from the demonstration. These skills can be observed as well as executed by the robot. The sequence of these skills observed forms the task model. The transition *a priori* skills are robot capabilities that are not observable but are required to execute the task.
These are used to control or execute the transitions in the task model. To execute an inferred task, the task has to be represented in a sequence of executable robot skills. The task execution sequence is developed by combining the task model and the transition *a priori* skills.

The observable *a priori* skills can be expanded based on sensors used for observing the demonstration. This provides a scalable method to expand the observable skill database. Using sensorized gloves for force-based gestures as shown in [24] provides additional informations for the demonstrations which can be used to extend the observable *a priori* skill database.

Examples of observable *a priori* skills are moving, setting specific joint positions, opening/closing the end effector, transporting an object (the movement with the object held in the end effector). Examples of transition *a priori* skills are picking, visual servoing, scooping, and placing. The robot has a pre-programmed database of these *a priori* skills. This database is queried for both the Human Task Inference and Robot Task Planning process.

GLOVEBOX SAFETY MEASURE THROUGH HARDWARE IMPLEMENTATION

Super Baxter enabled with PbD is susceptible to improper demonstrations or false task interpretation, and might result in breaking the glovebox. To ensure the safety of the glovebox, a fail-safe hardware joint limits are implemented on Super Baxter. These hardware joint limits are used to restrain the operational range of each arm using mechanical buffers, enabling the safe operations inside the glovebox. The joint limits are derived from exhaustive offline simulations.

Methodology

Mechanical limitation on the joints angles are defined in consideration with the position of Super Baxter inside glovebox. All possible trajectories within the joint limits, required for the task completion, are to be verified such that no part of the robot can hit the glovebox enclosure. An exhaustive offline simulation is proposed to determine joint limits and evaluate multiple scenarios before the physical implementation.

Initially, the simulation procedure is demonstrated with manipulators on a planar workspace, followed by 3D workspace. In a planar workspace, target points, P_i, can be objectively determined by task locations (x_i, y_i) and tooltip orientations (ϕ_i). Each target point has J sets of joint angle (θ) solutions, m_{ij} as represented in eq. (3), and is computed by solving Inverse Kinematics $\left(IK(P_i)\right)$, which satisfy the following equations:

$$P_i = (x_i, y_i, \phi_i) = \left(\Sigma_{q=1}^{Q} L_q \cos\left(\theta_q\right), \Sigma_{q=1}^{Q} L_q \sin\left(\theta_q\right), \Sigma_{q=1}^{Q} \theta_q\right) \qquad \text{(Eq. 1)}$$

$$IK(P_i) \in \left\{m_{i1}, m_{i2}, \dots, m_{iJ}\right\} \qquad \text{(Eq. 2)}$$

$$m_{ij} = \left\{\theta_1, \theta_2, \dots, \theta_q\right\} \qquad \text{(Eq. 3)}$$

Where q is the number of independent joint and θ is constrained by the respective motor limits

Usually, there are multiple inverse kinematic solutions for a given target location. A set of arm configurations covering all target locations provides the joint limit range with upper/lower bound. The joint limit range is then used in exhaustive search via “what-if” analysis for possible collisions to the glovebox.

For a 2-DOF arm, the target points in the robot workspace and the two joint configurations to reach target points are shown in Fig. 6a and Fig. 6b. For simplicity, we consider a task that requires to move around four target positions. Green lines indicate elbow-up arm configuration, and red lines indicate elbow-down arm configuration to reach target positions. From Fig. 6a, the workspace enclosure of the 2-DOF arm is represented as a circle. The graph clearly shows that the 2-DOF manipulator arm has high chance of hitting the glovebox enclosure; the cyan circle is not contained within glovebox perimeter (thick red line). After mechanical limit is applied, the reach of each link and the end effector will be contained within the glovebox, thus eliminate possibility of breaking through the wall, as shown in Fig. 6b.

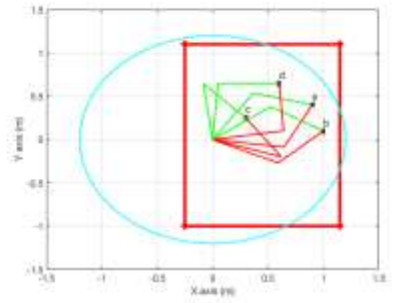

Fig. 6a. 2-DOF planar with joint solutions to reach 4 destination points

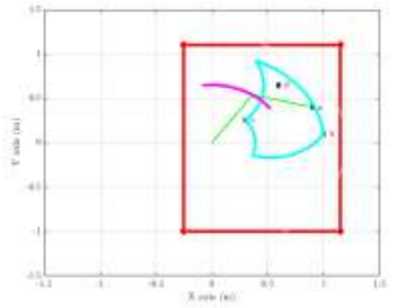

Fig. 6b. Workspace of each link with joint angle limits

In some case of different target positions, elbow-up arm configuration can hit the wall. Fig. 7a shows the target positions moved with an offset in upper-left direction, which prevent the elbow-up arm configuration to safely operate in the workspace, as the elbow link penetrates through the wall. So, the elbow-down arm configuration is used to determine the joint angle limits to enable safe operations. This configuration will still ensure that the target position can be reached and not penetrate through glovebox enclosure, but it will result in different workspace enclosure comparing to elbow up configuration (as shown in Fig. 7b.

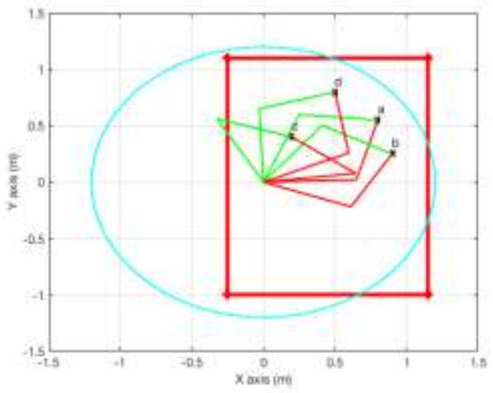

Fig. 7a. Target locations change, arm movement is restricted to elbow-down configuration

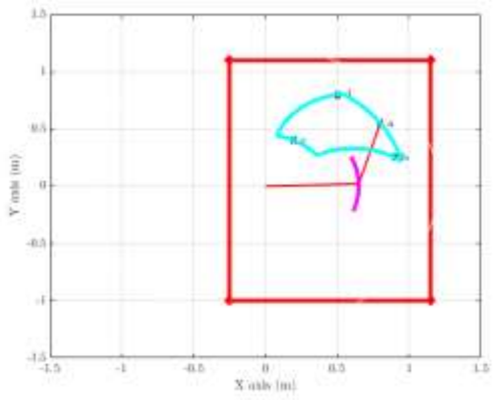

Fig. 7b. Workspace of each link with joint angle limits

To demonstrate the scalability of the above mentioned simulation approach, the scenario for 3-DOF manipulator in planar workspace is discussed further. In general, the 3-DOF manipulator has infinitely many joint configurations for a target position. Therefore, choosing the appropriate orientation out of infinite solutions is critical and task specific.

The equation of 3-DOF manipulator in planar workspace case is,

$$P_i = (x_i, y_i, \phi_i) = \left(\Sigma_{q=1}^{3} L_q \cos(\theta_q), \Sigma_{q=1}^{3} L_q \sin(\theta_q), \Sigma_{q=1}^{3} \theta_q\right) \quad \text{(Eq. 4)}$$

Let’s consider the pipetting task to simulate the 3-DOF manipulator to determine the joint angle limits for safe operations inside the glovebox.

In pipetting task, the robot must be able to move in a straight trajectory and maintain orientation of the tool’s endpoint. The endpoints’ coordinates and orientations are empirically chosen, as shown in Fig. 8a. The reach of each joint is shown in Fig. 8b.

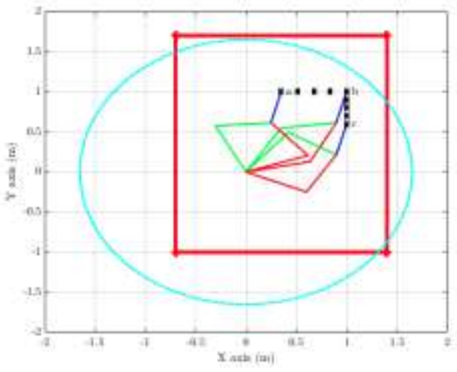

Fig. 8a. 3-DOF planar with joint solution to reach destination points

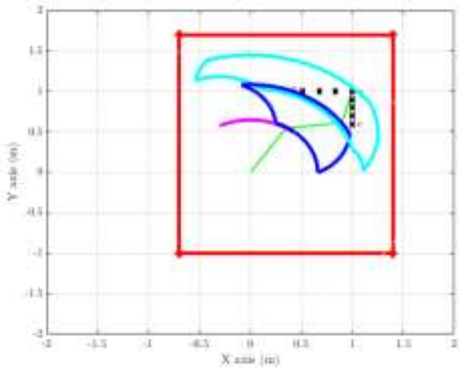

Fig. 8b. Workspace of each link with joint angle limits

A common issue with serial chain manipulators is with the manufacturing limit on the joint motors and could contribute to collision. Once the robot reaches the motor joint limit, it will move the joint to the opposite direction and attempt to reach the designated joint angle. We will demonstrate the effect of joint motor limit in the following Fig. 9a. and Fig. 9b., which compares endpoint trajectory of the 3-DOF manipulator with and without the motor joint limit. Without the joint motor limit, the arm manipulator will be able to move from point a to b without any issue with operational range on joint1 (base joint) = 63.64 degrees, joint2 = 43.94 degrees, and joint3 = 24.99 degrees. On the other hand, with motor joint limit, the joint operational range has to increase to cope with the motor limit and thus will enlarge the workspace enclosure with operational range on joint1 (base joint) = 63.64 degrees, joint2 = 343.6083 degrees, and joint3 = 24.99 degrees.

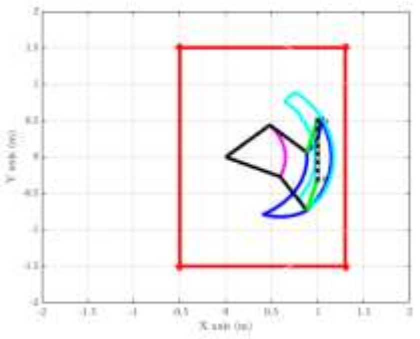

Fig. 9a. 3R planar following a straight line trajectory without motor joint limits.

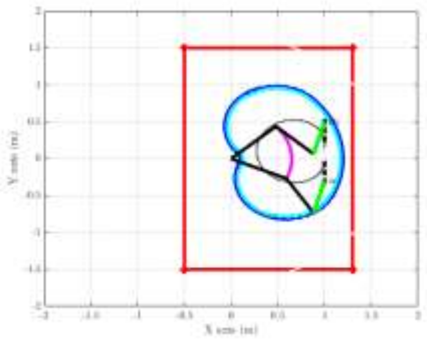

Fig. 9b. 3R planar following a straight line trajectory with motor joint limit

The similar idea can be implemented on Super Baxter, which has 14 degree of freedom, which can be developed based on the same method and principle, only different is that it takes more computation time to iterate through all the possible scenario.

EXPERIMENTS

Applied Gesture Based Programming to Perform Tasks

The usual glovebox tasks performed in the nuclear facilities are packaging of nuclear materials such as dry plutonium powder into containers, and pick & place the containers onto and off conveyor belts while transferring them. With that in mind, we design our simulated glovebox environment with the pick and place task from the conveyor belt next to Super Baxter. In our scenario, the demonstrator performs a pick and place task. In this task, the demonstrator picks up blocks from the conveyor and put them out of the conveyor belt (Fig. 14). During the human demonstration of the task, the object and hand movements are used to infer the task. The information about this task is then used for generating the task model from the task representation obtained by the inference phase. Super Baxter then learns the pick and place task from the demonstration and performs it (Fig. 15). The Super Baxter uses visual servoing to control the end-effector to perform the task.

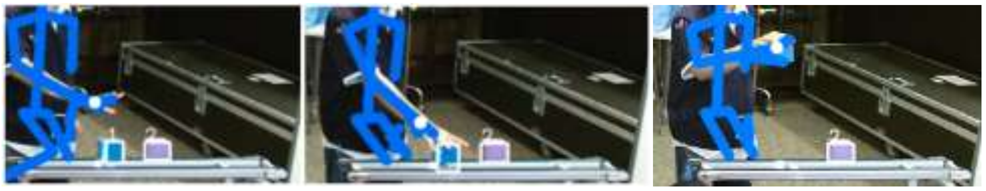
Fig. 14. Using the skeleton data (specifically hand position) and the position of the object to infer tasks such as approach and transport.

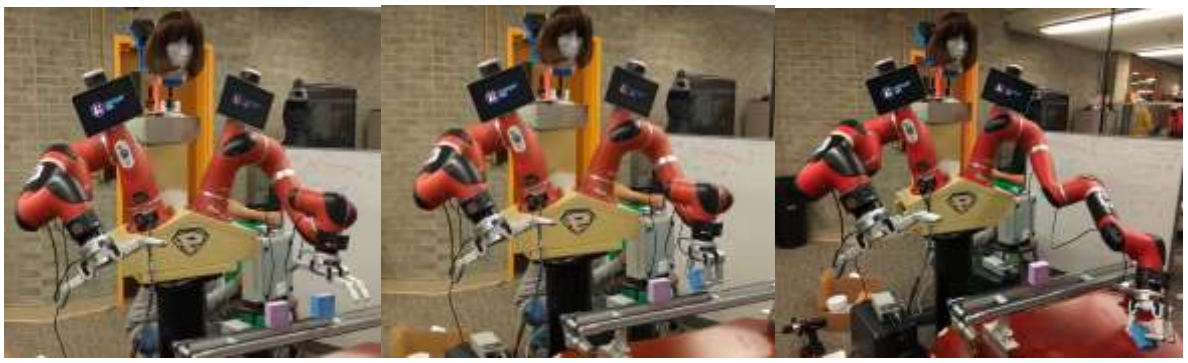
Fig. 15. Pick and place task performed by Super Baxter with visual servoing

In our next scenario, the demonstrator performs a scooping task. In this task, the demonstrator picks up the shovel and use them to scoop up metal pellets. As described previously in Robot Task Planning section, Super Baxter performs the scooping and un-scooping by inferring task learned from human demonstrator (Fig. 16.), and mapped the task to the robot workspace (Fig. 17.).

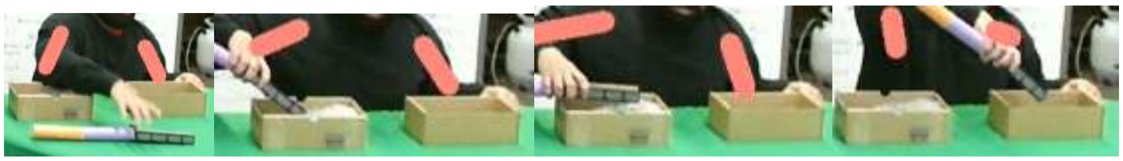
Fig. 16. Scoop / unscoop task performed by demonstrator

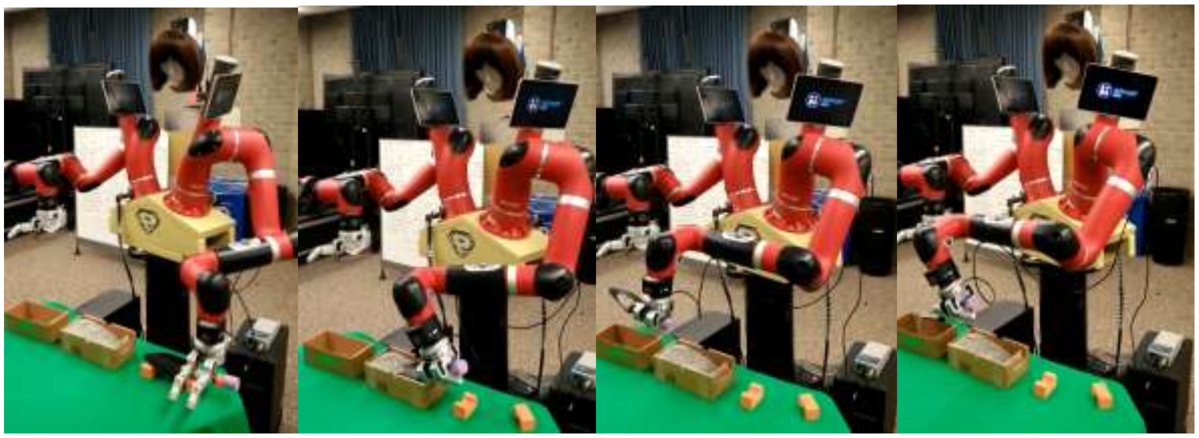
Fig. 17. Scoop / unscoop task performed by Super Baxter

CONCLUSION

This paper presents the preliminary work on teaching tasks to a robot through gesture based programming. Here we show feasibility of tasks similar to those performed in a glovebox. Super Baxter can be used to safely manipulate hazardous material inside the glovebox. Super Baxter can learn tasks to be performed by observing human gestures during a demonstration outside the glovebox. This eliminates the risks of operating in the glovebox. The GbP approach also enables the worker who has implicit knowledge of the task to easily program the robot to perform the task outside the glovebox without needing any expertise in programming. Safety of the glovebox is achieved by incorporating hardware safety measures on joint limits. These limits are obtained by exhaustive offline simulation. The joint limits computed from the simulation will be used to implement mechanical limit on the operational range of each arm.

FUTURE WORK

We aim to evaluate Super Baxter's performance using a glovebox. Additionally, we will develop a method for improving task performance by incorporating user inputs during task execution by Super Baxter. The user can provide more demonstrations of sub-parts of the task or can use audio cues to improve the task performance. This is similar to coaching. This provides a means for workers to improve the efficiency of task execution by providing feedback through communication or gesture cues.

REFERENCES

1. Occupational safety and health standards: Occupational health and environmental control (Standard No. 1926.1101) Occupational Safety and Health Administration. U.S. Department of Labor. (1970). https://www.osha.gov/pls/oshaweb/owadisp.show_document?p_id=10862&p_table=standards
2. Cournoyer, M. E., Garcia, V. E., Gallegos, U. F., & Wilburn, D. W. (2011). Investigation of injury/illness data at a nuclear facility. *Journal of Chemical Health and Safety*, *18*(5), 17-25.
3. Cournoyer, M. E., Kleinsteuber, J. F., Garcia, V. E., Wilburn, D. W., George, G. L., & Blask, C. L. (2011). Safety observation contributions to a glovebox safety program. *Journal of Chemical Health and Safety*, *18*(5), 43-55.
4. Trevelyan, J., Hamel, W. R., & Kang, S. C. (2016). Robotics in hazardous applications. In *Springer handbook of robotics* (pp. 1521-1548). Springer International Publishing.
5. Thompson, J. L. (2014). *Redesigning the human-robot interface: intuitive teleoperation of anthropomorphic robots* (Doctoral dissertation).
6. Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, *57*(5), 469-483.

7. Kory Westlund, J., Jeong, S., Park, H. W., Ronfard, S., Adhikari, A., Harris, P. L., DeSteno, D., & Breazeal, C. (2017). Flat versus expressive storytelling: young children's learning and retention of a social robot's narrative. Frontiers in Human Neuroscience, 11.
8. Cui, Y., Voyles, R. M., Nawrocki, R. A., & Jiang, G. (2014). Morphing bus: A new paradigm in peripheral interconnect bus. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, *4*(2), 341-351.
9. Delaunay, F. C. (2016). A Retro-Projected Robotic Head for Social Human-Robot Interaction.
10. Mizanoor, R. S., Spencer, D. A., Wang, X., & Wang, Y. (2014). Dynamic Emotion-Based Human-Robot Collaborative Assembly in Manufacturing: The Preliminary Concepts. In *Workshop on Human-Robot Collaboration for Industrial Manufacturing at RSS'14*.
11. Mollahosseini, A., Graitzer, G., Borts, E., Conyers, S., Voyles, R. M., Cole, R., & Mahoor, M. H. (2014). Expressionbot: An emotive lifelike robotic face for face-to-face communication. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on* (pp. 1098-1103). IEEE.
12. Colburn, A., Cohen, M. F., & Drucker, S. (2000). The role of eye gaze in avatar mediated conversational interfaces. *Sketches and Applications, Siggraph'00*.
13. Althloothi, S. R., Mahoor, M. H., & Voyles, R. M. (2012). Fitting distal limb segments for accurate skeletonization in human action recognition. Journal of Ambient Intelligence and Smart Environments, 4(2), 107-121.
14. Hasani, B., & Mahoor, M. H. (2017). Facial Expression Recognition Using Enhanced Deep 3D Convolutional Neural Networks. arXiv preprint arXiv:1705.07871.
15. Cabrera, M., Novak, K., Foti, D., Voyles, R., & Wachs, J. (2017). What makes a gesture a gesture? Neural signatures involved in gesture recognition. arXiv preprint arXiv:1701.05921.
16. Jacob, M. G., Li, Y. T., & Wachs, J. P. (2012, March). Gestonurse: a multimodal robotic scrub nurse. In Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on (pp. 153-154). IEEE.
17. Voyles, R. M., & Khosla, P. K. (1995, August). Tactile gestures for human/robot interaction. In Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on (Vol. 3, pp. 7-13). IEEE.
18. Ward, W., Cole, R., Bolaños, D., Buchenroth-Martin, C., Svirsky, E., Vuuren, S. V., ... & Becker, L. (2011). My science tutor: A conversational multimedia virtual tutor for elementary school science. ACM Transactions on Speech and Language Processing (TSLP), 7(4), 18.
19. Voyles, R. M., & Khosla, P. K. (2001). A multi-agent system for programming robots by human demonstration. Integrated Computer-Aided Engineering, 8(1), 59-67.
20. Dufay, B., & Latombe, J. C. (1984). An approach to automatic robot programming based on inductive learning. *The International journal of robotics research*, *3*(4), 3-20.
21. Muench, S., Kreuziger, J., Kaiser, M., & Dillman, R. (1994). Robot programming by demonstration (rpd)-using machine learning and user interaction methods for the development of easy and comfortable robot programming systems. In *Proceedings of the International Symposium on Industrial Robots* (Vol. 25, pp. 685-685). INTERNATIONAL FEDERATION OF ROBOTICS, & ROBOTIC INDUSTRIES.
22. Argall, B. D., Browning, B., & Veloso, M. M. (2011). Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot. Robotics and Autonomous Systems, 59(3), 243-255.
23. Derimis, Y., & Hayes, G. (2002). Imitations as a dual-route process featuring predictive and learning components: a biologically plausible computational model. *Imitation in animals and artifacts*, 327-361.
24. Voyles, R. M., & Khosla, P. (1996). Gesture-Based Programming, Part 1: A Multi-Agent Approach.
25. Kuniyoshi, Y., Inaba, M., & Inoue, H. (1994). Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE transactions on robotics and automation*, *10*(6), 799-822.

26. Inamura, T., Kojo, N., & Inaba, M. (2006). Situation recognition and behavior induction based on geometric symbol representation of multimodal sensorimotor patterns. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on* (pp. 5147-5152). IEEE.
27. Alissandrakis, A., Nehaniv, C. L., & Dautenhahn, K. (2002). Imitation with ALICE: Learning to imitate corresponding actions across dissimilar embodiments. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *32*(4), 482-496.

ACKNOWLEDGMENTS

This work is supported, in part, by the DOE/EM Science of Safety and by the NSF Center on RObots and SEnsors for the HUman well-Being (RoSe-HUB) through grants CNS-1439717 and CNS-1427872. Additional ideas were gleaned through support from NSF grants OISE-1550326, IIS-1752902 and IIS-1743285.