Formal Verification of a Fully Automated Out-of-Plane Cell Injection System

Iram Tariq Bhatti and Osman Hasan
School of Electrical Engineering and Computer Science (SEECS),
National University of Sciences and Technology (NUST), Islamabad, Pakistan.
(iram.tariq, osman.hasan)@seecs.nust.edu.pk

***Abstract*—Cell injection is a procedure in cell biology where a small volume of substance is injected into a specific location inside the cell. The overall success of the procedure in a fully automated cell injection systems mainly relies on the accurate path planning of the micro injector and the amount of forces applied to the cell at the time of injection. Traditionally, fully automated systems are analyzed using simulation, which is inherently non-exhaustive and incomplete in terms of finding potential system failures that might arise during operations. In this paper, we present a probabilistic model to analyze the functional correctness and performance of a fully automated out-of-plane cell injection system using the PRISM model checker. We use our model to verify an existing fully automated cell injection system and certain discrepancies are identified.**

***Index Terms*—Robotic Cell Injection System, Formal Verification, Probabilistic Model Checking, PRISM.**

I. Introduction

In biological cell injection, small quantities of substances, like, sperms, proteins, bio-molecules or genes, are injected into adherent or suspended cells for various medical applications ranging from drug development [1], in-vitro fertilization (IVF) [2], intracytoplasmic sperm injection (ISCI) [3], to gene injection [4] and others. For example, in IVF treatment, sperms are injected into a matured egg for curing infertility. Similarly, new medicines are developed through drug injection into a cell so that its outcome can be observed.

Fully automated cell injection systems can do cell injection automatically and allow batch processing contrary to the conventional approaches that are dependent on a skilled operator to inject material inside the cell and thus have lower success rates. The position of the cell to be injected is identified by using image processing techniques, then the control algorithms in the robotic manipulators automatically move the micro injector towards the cell for injection. However, to ensure high injection rates, the design parameters of these systems, like orientation of the coordinate frames, movement of the injection manipulator, microscope and digital cameras, have to be accurately modeled and analyzed to achieve accurate force control and precise motion of the micro injector [5]. A small modeling error in the movement and orientation of these components can cause the micro injector to penetrate the cell at an inappropriate location. Similarly, excessive force may damage the cell membrane [6] or an insufficient force may not allow the micro injector to cross the cell boundary [6].

Traditionally, the reliability of automated cell injection systems is ensured by techniques, like paper-and-pencil proofs, simulation and/or experimentation. However, paper-and-pencil proof based analysis can be human-error prone and is not scalable for large and complex models. On the other hand, simulation and experimentation is scalable but is based on sampling and thus cannot cover all the possible cases. To overcome the above-mentioned limitations of traditional analysis techniques, formal methods have been recently proposed [7] [8] to analyze fully automated cell injection systems.

Sardar et al. [8] formally modeled a fully automated cell injection system [6] as a Markov chain. However, the focus of his work was only on the formal modeling and no verification results were reported. Rashid et al. [7] used higher-order logic theorem proving for the verification of fully automated cell injection systems. In particular, they formalized the interrelationship of various coordinate frames present in the system to capture the dynamical behavior of these systems in the form of a mathematical relationship, which is then used to deduce properties about the force control and motion planning of the micro injector. Due to the undecidable nature of higher-order logic, the reasoning process involved explicit manual guidance. Moreover, the formal models used in this approach ignore the impact of noise and several other randomized behaviors.

We propose to use probabilistic model checking to ascertain accurate path planning of the micro injector and the application of ample force. In particular, we present a generic formal probabilistic model for a fully automated out-of-plane cell injection system and identify the formal properties that can be verified to ensure the desired characteristics of the given system. Moreover, to illustrate the utilization and practical effectiveness of the proposed approach, we formally verify an existing fully automated cell injection system developed by Huang et al. [5], which is an extension of the system that was formally modeled by Sardar et al. [8]. Based on our verification results, we identified certain anomalies in the values of the system damping and viscous friction effects. We thus proposed new system values, which leads to the correct motion of the micro injector towards the injected cell.

II. Preliminaries

A. Model Checking and PRISM

In model checking, the behavior of the system that needs to be verified is modeled as a state-space and its intended properties are modeled in temporal logic. The model checker then automatically and exhaustively checks if the given properties hold for the given model. In case of failure, an error trace is generated by the tool. Model checking suffers from the state-space explosion problem, i.e., when the state-space of the given system exceeds the available memory resources and thus the model fails to build. This problem is usually resolved by using an abstract and less complex model. Bounded and symbolic model checking [9] also allow us to handle the problems related to insufficient and less computational resources. Systems that exhibit random behavior are modeled and formally analyzed using probabilistic model checking strategies [10]. The behavior of these systems can be represented as a Markov chain where switching between states is done based on the likelihood of occurrence of events.

PRISM [11] is a widely used tool for probabilistic model checking. It supports four different probabilistic models [12], i.e., discrete time Markov chains (DTMCs), continuous time Markov chains (CTMCs), Markov decision processes, probabilistic automata (PAs) and probabilistic timed automata (PTAs). Each model allows verification of certain types of properties, like transition and steady-state probabilities, cost and reward structures etc.

A PRISM model primarily consists of modules and variables, where system behavior is defined within one or more modules having local variables and guard commands. The next variable states in the system are decided based on the present state of variables and the guard commands and its probability of occurrence. The syntax of a simplest PRISM command is: `[] guard -> prob_1:update_variable_1 + ... + prob_n:update_variable_n`. The probabilities of outgoing transitions from a state has to be 1.

We can verify properties specified in probabilistic computational temporal logic (PCTL) for DTMC based models using PRISM. The most commonly used operator for property specification is the `P` operator, which provides the probability of occurrence of a particular event in any given path in the system that satisfies the property `path_prop` and checking it over a certain `bound`, i.e., `P bound [path_prop]`.

Quantitative analysis is generally performed to find the actual likelihood of observing a particular behavior in the system following different paths within the model starting from a given state. Path property is written using temporal operators, which include `F` for eventually aka future, `G` for always aka global, `X` for next and `U` for until. `Labels` are Boolean operators that provide an easy way for property checking over a set of states in the system.

B. Automated Cell injection systems

Generally, a fully automated cell injection system has three basic modules, i.e., an *executive* module, which constitutes the working plate, the positioning table and the micro injector, a *sensory* module, which comprises of a vision system with four sub components, i.e., an optical microscope, charged coupled device (CCD) camera, peripheral component interconnect (PCI) image capture card and a *control* module, which includes the host computer with a motion control device, as shown in Fig. 1. Successful injection is performed when all the modules communicate with each other.

The configuration of an automated cell injection system is depicted in Fig. 1. The stage, which comprises of table and working plate, has $O-XYZ$ coordinate frame axis, where the origin O of these coordinates lies at the center of the working plate and Z is the optical axis of the microscope. In the same way, $O_c - X_cY_cZ_c$ represents the camera coordinates, where O_c is situated at the center of the microscope camera frame. The image plane coordinates are represented by $O_i - uv$, where O_i is the origin and the uv axis is perpendicular to the optical axis of the image frame.

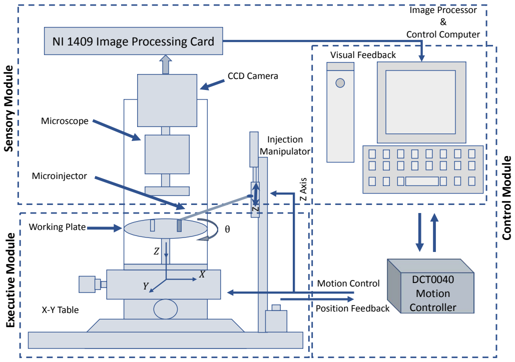

Fig. 1: Cell Injection System Components

The interrelationship between the $[u, v, Z, \theta]^T$ of the image frame to the $[X, Y, Z, \theta]^T$ of the stage frame for 4DOF automated out-of-plane cell injection system [13] is:

$$\begin{bmatrix} u \\ v \\ Z \\ \theta \end{bmatrix} = \begin{bmatrix} f_x \cos\alpha & f_x \sin\alpha & 0 & 0 \\ -f_y \sin\alpha & f_y \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ \theta \end{bmatrix} + \begin{bmatrix} f_x d_x \\ f_y d_y \\ 0 \\ 0 \end{bmatrix} \quad (1)$$

where α is the angle between the two frames. Similarly, f_x, f_y, d_x and d_y are also constant parameters. The display resolution of the vision system is expressed by $f_x = \lambda/\delta_u$ and $f_y = \lambda/\delta_v$ in XY directions where λ is the microscope magnification factor, δ_u and δ_y are the u-axis and v-axis intervals between CCD pixels, respectively.

The generalized Lagrange equation of motion for the motion stage is given by Equation 2, where $q = [u, v, Z, \theta]^T$ is the generalized coordinates, M is the inertia matrix term, N models the system damping and the viscous friction effects, G represents the gravitational force vector term, J is the Jacobin matrix and τ represents the system torque.

$$M\ddot{q} + N\dot{q} + G = \tau + \begin{bmatrix} J^T \\ 0 \end{bmatrix} \quad (2)$$

III. Proposed Methodology

A. Probabilistic Formal Model

The first step in the proposed methodology, depicted in Fig. 2, for formal probabilistic analysis of fully automated out-of-plane cell injection systems is to develop a formal model for its system behavior. This is done by selecting an appropriate Markovian model for the given system. We propose to use a DTMC [12] as the impact of randomness due to noise and unpredictable surroundings can be captured through this type

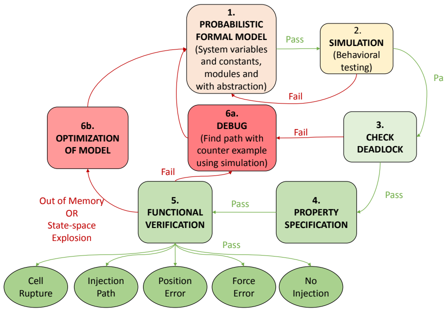
1. PROBABILISTIC FORMAL MODEL (System variables and constants, modules and with abstraction)
Pass
2. SIMULATION (Behavioral testing)
Pass
Fail
3. CHECK DEADLOCK
Fail
6a. DEBUG (Find path with counter example using simulation)
6b. OPTIMIZATION OF MODEL
Pass
4. PROPERTY SPECIFICATION
Pass
5. FUNCTIONAL VERIFICATION
Fail
Out of Memory OR State-space Explosion
Pass
Cell Rupture
Injection Path
Position Error
Force Error
No Injection

Fig. 2: Proposed Methodology

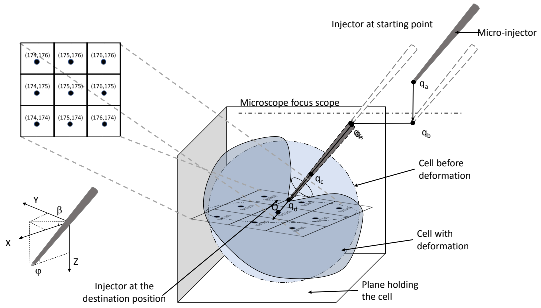

Fig. 3: Cell Position Coordinates

of a model in a very straightforward manner. The proposed modeling is based on a modular approach. The *controller* modules computes the force and the torque, the *plant* module is used to update position, the *noise* module represents the behavior of randomness in the system and the *desired* module is used to provide the reference values of the micro injector motion and accompanying force error reduction in the closed-loop system design.

1) Identification of Cell Position: In the out-of-plane automated cell injection, the first step is to bring the micro injector in the microscope's focal plane and adjust its position such that its tip points towards the center of the cell. We propose to model the cell position as a dynamic variable specified by the user at the time of building the model.

2) Stages of Cell Injection: The injection procedure is broadly divided into several stages namely, *pre-piercing*, *piercing* and *injection*. The pre-piercing stage represents the phase when the micro injector has not yet touched the cell boundary. The piercing stage starts when the micro injector begins to creates a dimple in the cell wall in order to break through it. The injection stage commences when the micro injector is physically inside the cell causing no harm to its structure. The micro injector then drops the substance in the cell and subsequently leaves it.

3) Optimization of the Model: In the model optimization, we abstract the model so that no interesting feature is lost and the model is also reasonable enough to cater for the state-space explosion problem. The main parameters that we can abstract in this step includes the start and end points of the micro injector motion, the step size and the noise level.

4) Simulation: Once the initial model is ready, the inbuilt PRISM simulator is used to check its functional correctness. The simulator automatically selects random test vectors and its interactive interface allows us to easily view the state-space to detect any possible anomalies in the model. This step allows us to identify any human errors or inappropriate abstractions through debugging the design. The model is then updated to fix any issues that are detected in the model. This step is quite vital before rigorous verification as it is quite quick and straightforward in debugging the model.

5) Deadlock Freeness: A deadlock represents an unwanted scenario during injection where the considered system halts its execution. Model checking is apt in detecting possible error traces that might lead to a deadlock. In PRISM, deadlock is a built-in label, and both simulation and the property `E [F "deadlock"]` can be used to check the absence of deadlock along any path in the model. If a deadlock is found in any path during simulation, the root cause can be debugged to eliminate its effect. If the root cause is because of some modeling errors then the model is refined to resolve it, otherwise, if the source of deadlock resides in the functional behavior of the system, the issue can be resolved by revising the design itself.

6) Property Specification and Functional Verification: The next stage comprises of specifying and verifying the behavioral properties for the considered system. In this regard, we propose estimating the likelihood of poor injection, possibility of cell rupture during the cell operation, probability of force error exceeding a certain threshold and the chances of motion error and verifying the accuracy of the model by checking the correct execution order of the injection stages. These properties can be represented in PRISM as follows:

1) $\text{P}_{\text{poor\_injection}}$ =? [F poor_injection]
2) $\text{P}_{\text{cell\_rupture}}$ =? [F cell_rupture]
3) $\text{P}_{\text{force\_error}}$ =? [F force_error $>$ bound]
4) $\text{P}_{\text{position\_error}}$ =? [F position_error $>$ bound]
5) $\text{P}_{\text{stages\_of\_injection}}$ =? [F stage1 U stage2 U stage3]

If out-of-memory or state-space explosion problems are encountered during verification then the model is re-examined for feasible options for variable reordering, value optimization or bounds adjustment in the initial or the final values by going back to the optimization step. In case of failure, the counter example is critically viewed in the simulator and the source of the failure is then identified and rectified.

IV. Formal Analysis of a Cell Injection System

In this section, we illustrate the usefulness of the proposed methodology by applying it to analyze a real-world cell injection system [5], as shown in Fig. 3. In the case of out-of-plane injection, the micro injector is initially brought in the focal view of the microscope by moving vertically from position q_a to q_b, then it is moved horizontally to position q_s, reffered as the starting position of the injection process, such that the tip of the micro injector points towards the center of the cell. From this point onwards, the diagonal motion of the injector is performed towards the center of the cell. As the injector is viewed under the microscope, so the actual three

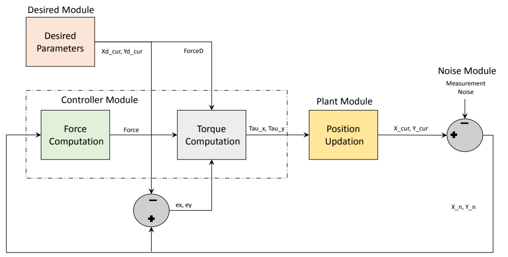

Fig. 4: Proposed Close-loop System Model

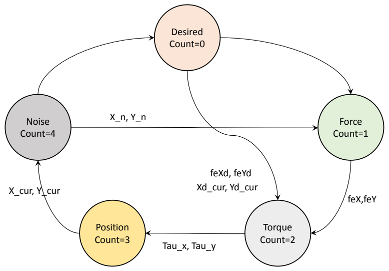

Fig. 5: Proposed System

dimensional motion is viewed as two dimensional motion. We propose that the relationships for the force and the torque between the cell and the micro injector, the impedance-control, and the error reduction for the closed-loop system, given in [5], to be directly used in our models. In the automated out-of-plane cell injection system [14], the position of the micro injector and the cell on the rotary plate is identified by using image processing techniques. Once the cell coordinates are successfully identified the micro injector moves towards it by traversing the desired motion. Similarly, we model the interaction of cell with the micro injector as user defined variables, i.e., `CellBoundaryX` and `CellBoundaryY` as illustrated by the example of a 3x3 matrix with values ranging between 174-176μm in Fig. 3.

In the considered system [5], the rotation angle between the image and the stage frame, α, is 0°. The magnification of microscope's objective, λ, is 30. The angle between the micro injector and the X-axis, β, is 45°, whereas the angle between the micro injector and the Z-axis, φ, is 35.26°. The initial modeling values of M, the inertia matrix, N, the system damping and the viscous friction effects, G, the gravitational force vector term are taken from [13]. In the PRISM model, the magnification factor λ is used with the PULNiX TM-6701AN progressive scan CCD camera and the values for image frame δ_u and δ_v are 9μm.

A. System Modules

The interaction of the of four basic modules, as given in Section III, leads to a closed-loop system model as shown in Fig. 4 and the flow within each iteration of the closed-loop system is elaborated in Fig. 5. The variable `count` starts from the desired module where the reference data of the desired position and the force variables `feXd`, `feYd`, `Xd_cur` and `Yd_cur` is available. The controller module then computes the forces based on the distance that the micro injector traveled inside the cell. This force value is then decomposed into `feX` and `feY` and is sent to the second component of the controller module for the torque calculations. The torque is computed by using the actual and the desired force values. The generated torque values `Tau_x` and `Tau_y` are then used by the plant module to update the coordinates of the micro injector, i.e., variables `X_cur` and `Y_cur`. The environmental uncertainties in the form of the Gaussian noise are then added to the position variables of the micro injector `X_n` and `Y_n` inside the noise module. The loop keeps repeating until the micro injector is successfully injected in the cell.

1) Controller Module: The controller computes the values of the force and the torque in the system. Instead of the bio-membrane point load model [15], used in [8], the force in the controller module is computed based on the abstraction defined in [5], where the distance `s` of the micro injector inside the cell determines the force. In the formula `S_dist` [5] given below, `Xqc` and `Yqc` are the XY boundary points of the cell as entered by the user. The `X_n` and `Y_n` are the position coordinates of the micro injector after the addition of the Gaussian noise.

```
formula S_dist = ceil(pow(pow(pow(X_n - Xqc, 2) + pow(Y_n - Yqc, 2) +
                 (3.00213 ∗ (pow(X_n - Xqc, 2) + pow(Y_n - Yqc, 2))),
                 0.5));
```

The force values provided by `S_dist` are divided into four categorizes [5]. In the pre-piercing stage (condition 1), the micro injector and the cell are not in contact with each other and therefore the force is zero. In the piercing stage (condition 2-3), the injector starts creating a dimple inside the cell in an attempt to pierce through it, and thus requires some force. Finally, in the injection stage (condition 4), the micro injector has pierced the cell wall and the force is zero again.

```
 [] guard − > (Force' = 0);
[] guard − > (Force' = min(max(ceil(8.5714∗S_dist), 0), Force_max));
[] guard − > (Force' = min(max(ceil(0.02274∗pow(S_dist, 2) −
        0.09252∗S_dist + 95.31), 0), Force_max));
[] guard − > (Force' = 0);
```

The calculated force is then decomposed into its components using the torque equation [5], where the desired force is the last variable quantity as pointed out by Rashid et al. [7]. The resulted code with the bounded model checking is

```
 [] guard − > (Tau_x' = min(max(ceil(104.1412293∗Xd_cur
         − 50.12831775∗Xd_old1 + 2.12∗Xd_old2 − 108.5542293∗X_n
         + 50.42131775∗X_n_old1 − 6.424236∗feX + 43.748
         + 3.333333×10^6∗feXd), TauX_min), TauX_max));
[] guard − > (Tau_y' = min(max(ceil(53.59362672∗Yd_cur
       − 23.17593265∗Yd_old1 + 1.086∗Yd_old2 − 54.85762672∗Y_n
       + 23.35393265∗Y_n_old1 − 4.4098116∗feY + 23.168
       + 3.333333×10^6∗feYd), TauY_min), TauY_max));
```

2) Plant Module: The motion of the micro injector for the fully automated cell injection system for the out-of-plane injection is modeled using the Language equation of motion

[13]. Moreover, backward difference method [16] is used as proposed by Sardar et al. [8] to get the final equations for the micro injector position as represented in PRISM with bounded model checking for the considered system.

```
[] guard − > (X_cur' = min(max(ceil(0.1214256131∗X_n_old1 +
        0.8785743887∗X_n_old2 − 1.243265644 ×10^−7∗Tau_x +
        0.4144218815∗feX + 18.13012847), InitialPositionXY),
        MaxDistance));
[] guard − > (Y_cur' = min(max(ceil(0.14082278∗Y_n_old1 +
        0.8591772152∗Y_n_old2 − 2.373417722 ×10^−7∗Tau_y +
        0.7911392405∗feY + 18.32911392), InitialPositionXY),
        MaxDistance));
```

3) Noise Module: Noise can arise as a result of inaccuracies in the sensor being used, like calibration error, lifetime of equipment and fabrication errors [17]. Other random factors can be classified as internal or external disturbances depending upon the origin of the noise whether it lies inside or outside of the system. Internal disturbances represent the uncertainties that originates from the electromagnetic effects of the components in the system, the variability of the operating point, the variation in the process parameters and the distortion due to non-linear elements in the system [17]. External disturbances are caused by the environmental effects, such as change in the temperature and electromagnetic effects of the components in the surrounding, for instance, high temperature may lead to image degradation and may effect the calculation of the cell coordinates and its deformation values.

In the considered system, the Gaussian noise model is used with `NoisePercentage` level set to 0.01. The Gaussian noise is added only in the motion governing parameter of the micro injector as the system design allows the impact of the noise to be automatically translated to the force as well.

```
[] guard − > 0.2:(X_n' = min(max(ceil(X_cur − NoisePercentage∗X_cur),
             InitialPositionXY), MaxDistance)) +
             0.6:(X_n' = min(max(ceil(X_cur), InitialPositionXY),
             MaxDistance)) +
             0.2:(X_n' = min(max(ceil(X_cur + NoisePercentage∗X_cur),
             InitialPositionXY), MaxDistance));
[] guard − > 0.2:(Y_n' = min(max(ceil(Y_cur − NoisePercentage∗Y_cur),
             InitialPositionXY), MaxDistance)) +
             0.6:(Y_n' = min(max(ceil(Y_cur), InitialPositionXY),
             MaxDistance)) +
             0.2:(Y_n' = min(max(ceil(Y_cur + NoisePercentage∗Y_cur),
             InitialPositionXY), MaxDistance));
```

4) Desired Module: The fully automated out-of-plane cell injection system is designed as a closed-loop system to minimize the error in the system by continuously comparing the desired and actual values [5] of the force and the position.

B. Verification

In this section, we present the verification results of the proposed model of fully automated out-of-plane cell injection system [5] using PRISM. The results reported below are obtained by building the model with constants `StepSizeXY`=18, `InitialPositionXY`=100, `MaxDistance`=250, `CellBoundaryX`=174:176, `CellBoundaryY`=174:176 and `NoisePercentage`=0.01 with PRISM switches `-cuddmaxmem`=2048g, `-javamaxmem`=512g and `-ex`. Our model was found to be deadlock free.

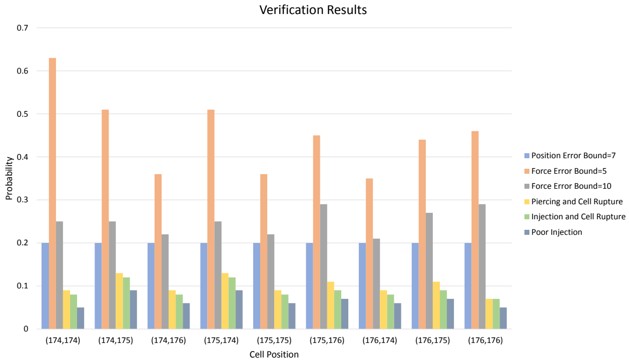

Fig. 6: Verification Results

1) Position Error: We verified the probability of the micro injector to follow the desired motion trajectory in order to safely inject material inside the cell. PRISM's inbuilt simulator shows that the individual motion profiles in both axes largely follow the desired motion profiles to a great extent. However, as the simulator is not exhaustive, we propose to do analysis through a position error property. We have defined `X_err` and `Y_err` labels in PRISM that use the percentage error formula for predicting the position error between the desired and the actual injector motion in the model.

```
label "Xaxis_err" = ((((Xd_cur - X_n)/Xd_cur) ∗ 100 > Bound) | (((Xd_cur
        − X_n)/Xd_cur) ∗ 100 < −Bound)) & (Xd_cur >
        InitialPositionXY) & (Xd_cur < MaxDistance) & (X_n >
        InitialPositionXY) & (X_n < MaxDistance);
label "Yaxis_err" = ((((Yd_cur - Y_n)/Yd_cur) ∗ 100 > Bound) | (((Yd_cur
        − Y_n)/Yd_cur) ∗ 100 < −Bound)) & (Yd_cur >
        InitialPositionXY) & (Yd_cur < MaxDistance) & (Y_n >
        InitialPositionXY) & (Y_n < MaxDistance);
```

The label returns true only for those states in the model where the bound of more than 7% is crossed. The verification of the property expressed below provides a combined probability of error as 0.2 while considering all combinations of the cell positions, as shown in Fig. 6.

```
P = ? [ F"Xaxis_err" & "Yaxis_err" & count = 2]
```

2) Force Error: Any drastic difference in the actual force values compared to the desired values at any point can damage the cell and thus can lead to failed injection. The force error property is defined by the use of `Force_err` label.

```
label "Force_err" = ((((ForceD - Force)/ForceD) ∗ 100 > Bound) | (((ForceD
        - Force)/ForceD) ∗ 100 < -Bound)) & (Force > 0) & (Force
        < Force_max) & (ForceD > 0) & (ForceD < Force_max) &
        (X_cur = Xd_cur | Y_cur = Yd_cur);
```

The final property shows the corresponding force error probability as around 0.2 with an error threshold of 10%, whereas if the threshold is tightened to 5% then the range of the reported error varies between 0.35 to 0.63. The force error results for both bounds are shown in Fig. 6.

```
P=? [ F “Force_err” & count = 2]
```

3) Stages of Injection: The micro injector moves through several phases during a fully automated cell injection procedure. This shows that the automated cell injection system has accurately identified the cell position and it is proceeding in the correct direction. Injection stage labels allow us to verify this property as follows:

```
label “pre_piercing” = (X_n <= Xqc | Y_n <= Yqc);
label “piercing” = (X_n > Xqc | Y_n > Yqc) & S_dist <= CellPunctured;
label “injection” = (X_n > Xqc | Y_n > Yqc) & S_dist > CellPunctured;
```

The verification shows that the micro injector follows each stage as it moves towards its destination and is ensured as

```
P=? [ F (“pre_piercing” U “piercing”) U “injection” ]
```

4) Chances of Cell Rupture: During execution, if the force accidentally exceeds a certain threshold value it may damage the cell resulting in a failed procedure. Thus, in the PRISM model, a threshold value of the force is defined and the probability of cell rupture is found during the piercing and the injection stages by using `cell_rupture` label.

```
label “cell_rupture” = (Force > Force_threshold);
```

The results show the reliability of the model as the chances of cell rupture are very slim during the piercing and injection stages as its values are below 0.15, as reported in Fig. 6.

```
P=? [ F “piercing” & “cell_rupture” ];
P=? [ F “injection” & “cell_rupture” ];
```

5) Probability of Poor Injection: The micro injector can successfully pierce the cell boundary when the force at the time of injection is greater than the minimum force needed at that instance. Thus, if the value of the applied force at the time of injection is below the desired value then the micro injector would not be able to penetrate inside the cell. The property with the associated labels is written in PRISM as

```
label “poor_injection” = (Force < ForceD) & (X_n > Xqc | Y_n > Yqc) &
                        S_dist = CellPunctured;
    P=? [ F “piercing” & “poor_injection” ];
```

The results, shown in Fig. 6, show that the success rates of the system under analysis is over 90%.

C. Discussion and Findings

During the verification process, certain discrepancies are identified in the considered system [5], like, the value of the system matrices, given in [13], when used in the proposed model caused the system to enter a deadlock state. Furthermore, the formal analysis showed that the values of system damping and viscous friction effects i.e., *N*, used in the original design [5] were inaccurate, as the movement of the micro injector in each axis contradicted with the desired orientation. We were able to identify the correct values as follows:

$$N_{xy} = \begin{bmatrix} \frac{43.748}{\dot{X}} + 0.2335\dot{X} - 4.533 & 0 \\ 0 & \frac{23.168}{\dot{Y}} - 0.1406\dot{Y} - 2.35 \end{bmatrix} kg.m/s \tag{3}$$

V. Conclusion

In cell injection procedures, an appropriate force is needed at the time of injection, otherwise cell may be damaged and the procedure can fail. Thus, controlling the force values is very critical. Similarly, the micro injector has to follow a specific path towards the targeted cell for injection. To rigorously ensure these constraints, we proposed a probabilistic formal model for a fully automated out-of-plane cell injection system and used it to ensure the safety and reliability of an actual system. The current model can be extended to compute operational rewards, determine the effect of randomness in the force and verifying the model with different combinations of step sizes and noise levels.

References

[1] T. Nakayama, H. Fujiwara, K. Tastumi, K. Fujita, T. Higuchi, and T. Mori, “A New Assisted Hatching Technique using a Piezo-micromanipulator,” *Fertility and Sterility*, vol. 69, no. 4, pp. 784–788, 1998.

[2] Y. Sun and B. J. Nelson, “Biological Cell Injection using an Autonomous Microrobotic System,” *Robotics Research*, vol. 21, no. 10-11, pp. 861–868, 2002.

[3] K. Yanagida, H. Katayose, H. Yazawa, Y. Kimura, K. Konnai, and A. Sato, “The Usefulness of a Piezo-micromanipulator in Intracytoplasmic Sperm Injection in Humans,” *Human Reproduction*, vol. 14, no. 2, pp. 448–453, 1999.

[4] J. Kuncova and P. Kallio, “Challenges in Capillary Pressure Microinjection,” in *Engineering in Medicine and Biology Society*, vol. 2. IEEE, 2004, pp. 4998–5001.

[5] H. Huang, D. Sun, J. K. Mills, W. J. Li, and S. H. Cheng, “Visual-based impedance control of out-of-plane cell injection systems,” *IEEE Transactions on Automation Science and Engineering*, vol. 6, no. 3, pp. 565–571, 2009.

[6] H. Huang, D. g. Sun, J. K. Mills, and W. J. Li, “A visual impedance force control of a robotic cell injection system,” in *Robotics and Biomimetics*. IEEE, 2006, pp. 233–238.

[7] A. Rashid and O. Hasan, “Formal analysis of robotic cell injection systems using theorem proving,” in *Design, Modeling, and Evaluation of Cyber Physical Systems*. Springer, 2017, pp. 127–141.

[8] M. U. Sardar and O. Hasan, “Towards Probabilistic Formal Modeling of Robotic Cell Injection Systems,” in *Models for Formal Analysis of Real Systems*, 2017, pp. 271–282.

[9] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, Y. Zhu *et al.*, “Bounded model checking.” *Advances in computers*, vol. 58, no. 11, pp. 117–148, 2003.

[10] H. Oldenkamp, “Probabilistic model checking: A comparison of tools,” Master’s thesis, University of Twente, Enschede, the Netherlands, 2007.

[11] “Prism model checker,” https://www.prismmodelchecker.org, 2019.

[12] V. G. Kulkarni, *Modeling and analysis of stochastic systems*. Chapman and Hall/CRC, 2016.

[13] H. Huang, “Towards automatic batch biomanipulation: study on robotic suspended cell injection system,” Ph.D. dissertation, City University of Hong Kong, Hong Kong, 2008.

[14] H. Huang, D. Sun, J. K. Mills, and S. H. Cheng, “Integrated vision and force control in suspended cell injection system: Towards automatic batch biomanipulation,” in *Robotics and Automation*. IEEE, 2008, pp. 3413–3418.

[15] Y. Sun, K.-T. Wan, K. P. Roberts, J. C. Bischof, and B. J. Nelson, “Mechanical Property Characterization of Mouse Zona Pellucida,” *IEEE Transactions on Nanobioscience*, vol. 2, no. 4, pp. 279–286, 2003.

[16] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-state and Time-dependent Problems*. SIAM, 2007, vol. 98.

[17] R. C. Dorf and R. H. Bishop, *Modern control systems*. Pearson, 2011.