Module 5A - Modeling and Simulation of 3D Robotic Printers Module

Omar Betancourt, Payton Goodrich, Emre Mengi

April 2021

Contents

Objectives: Modeling and simulation of robots for 3D food-printing applications
Prerequisite Knowledge: High school physics
Prerequisite Modules: 1A - Calculus, 1B - Linear Algebra, 1D - Differential Equations, 2B - Continuum Mechanics, 2C - Particle Dynamics, 3C - Generic Time Stepping, 4A - Genetic Algorithms, 4B - Gradient-based Optimization
Difficulty: Hard
Summary: A 3D printer is modeled and then optimized using basic physics and genetic algorithms.

1 Theory

The goals are to explore the following topics:

- Modeling the dynamics of a 3D printing robot
- Dynamics and thermodynamics of printer material
- Time-discretization
- Machine-Learning for system training/optimization

Rapid free-form printing of particle-functionalized materials is an integral component of 3D printing and additive manufacturing. Such processes consist of attaching a dispenser to a robot arm which then releases droplets of specialized mixtures in free-space that are deposited onto a substrate. These approaches are popular because they utilize widely-available, highly-programmable, robots. However, often the release of a complex mixture in free-space is somewhat imprecise, thus electrodynamic field control has been proposed as a possible remedy to enhance the precision of such processes. Specifically, electrodynamic control of the material is achieved by electrifying the released material in the presence of a prescribed ambient electrical field generated from the substrate, in order to guide it to the desired position. There are many components that comprise such a system, thus motivating the construction of a simulation tool framework assembling submodels of the:

- kinematics of the robot arm and the dispenser-printer head,
- electrodynamic and gravitational forces on the released material,
- dynamics of the released material and
- conductive, convective and radiative cooling of the media.

The modular system structure allows for easy replacement of submodels within the overall framework. Numerical simulations are undertaken to illustrate the overall system model, which is comprised of an assembly of submodels. Afterwards, a Machine Learning Algorithm (MLA) is developed to identify and optimize the proper system parameters which deliver a desired printed pattern. Specifically, an MLA is developed to ascertain the appropriate combination of robotic motion and electrical fields needed to create structures which would be difficult or impossible by purely mechanical means alone. Afterwards, extensions involving more detailed models are then provided, based on the Discrete Element Method.

1.1 Introduction

As the rapid digitalization of industry has been taking place, manufacturers have been scrambling to meld simulation software with emerging technologies based on embedded sensor technology, autonomous robotics, virtual reality (VR) and artificial intelligence (AI). Integrated advanced sensors, controls, simulation platforms and modeling has made next-generation advanced manufacturing, sometimes referred to as Industry 4.0, possible (Zohdi and Dornfeld [1] and Huang et al [2]). For the purposes of this chapter, we refer to this entire evolving area as "Advanced Manufacturing". Advanced Manufacturing requires the development of models and simulation tools that can be run rapidly for design purposes, which then naturally lend

themselves to optimization strategies such as Machine Learning Algorithms (MLAs), which require several hundred or thousand simulation runs during in an extremely short period of time.

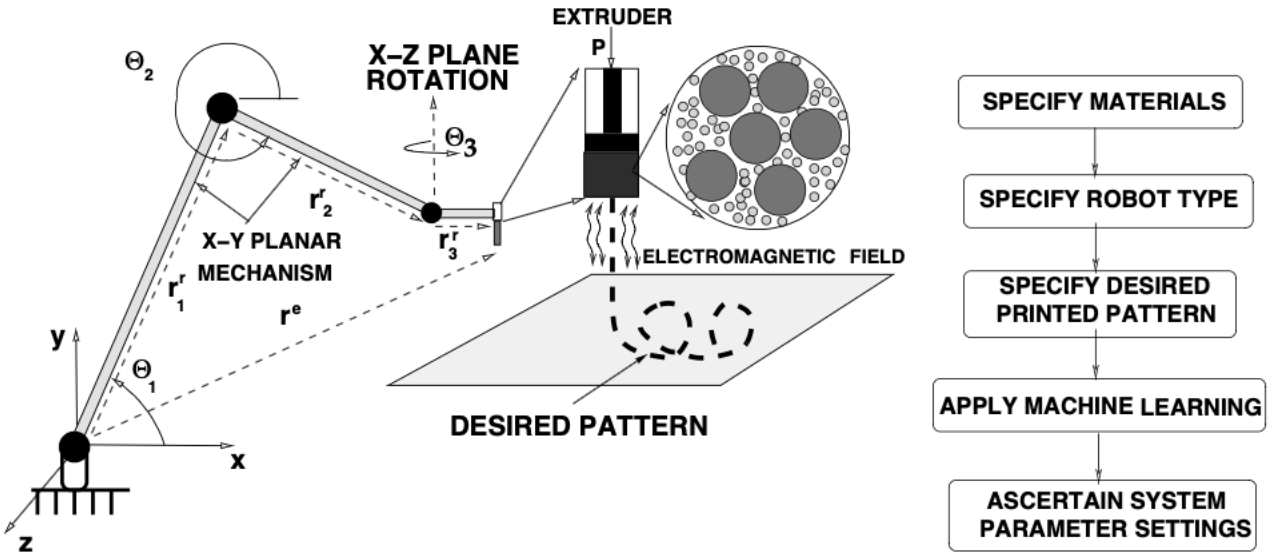

Figure 1.1: Model problem of a robot with a dispensing printer head.

Specialized MLAs, digital twin technology and complex materials are key factors in these processes, which have now become critical to industrialized economies. It is estimated that digitized manufacturing will generate $1 trillion in net global value over the next four years by streamlining design, manufacturing processes and managing supply chain risks. One type of paradigm that is useful to determine the parameters needed for multicomponent systems to properly function is Machine Learning, which has now really become critical for all forms of advanced manufacturing, such as 3D printing. A particular type of 3D printing that is becoming popular is free-form printing, which is comprised of a multi-axis robot arm equipped with a dispenser that dispenses droplets of specialized mixtures. However, this type of process has many design parameters. These approaches are popular because they utilize widely-available highly-programmable robots that have been developed for decades in other fields. However, often the release of a complex mixture in free-space is somewhat imprecise, thus electrical field control has been proposed as one possible remedy to enhance the precision of such processes. Specifically, electrical control of the material is achieved by electrifying the released material in the presence of a prescribed electric field generated on the substrate, in order to guide it to the desired position. There are many components that comprise such a system, thus motivating the construction of a simulation tool framework in this chapter assembling submodels of:

- The kinematics of the robot arm and the dispenser-printer head,
- The electrodynamic and gravitational forces on the released material,
- The dynamics of the released material and
- The conductive, convective and radiative cooling of the media.

The modular system structure allows for easy replacement of submodels within the overall framework. In this work, numerical simulations are undertaken to illustrate the overall system model, which is comprised of an assembly of submodels. An MLA is then developed to identify and optimize the proper system parameters which deliver a desired printed pattern. Specifically, an MLA is developed to ascertain the appropriate combination of robotic motion and electric fields needed to create structures which would be difficult or impossible by purely mechanical means alone. Afterwards, extensions using more detailed models are then provided, based on the Discrete Element Method.

1.2 A model problem: free-form robotic printer

As a model problem, (Figure 1.1), we consider a robot linkage, with a mounted dispenser, described via vector loops which are widely used in the robotics literature (for example, see Hunt [3], Hartenberg and Denavit [4], Howell [5], McCarthy [6, 7], Reuleaux [8], Sandor and Erdman [9], Slocum [10], Suh and Radcliffe [11] and Uicker et al. [12]). Links one and two are x-y planar, while link three is in the x-z plane. We assume that we can precisely control the angles and angular velocities of all of the links in the system. The position vector ($\boldsymbol{r}^d$) to the dispenser is given by

$$\boldsymbol{r}^d = \boldsymbol{r}_1^r + \boldsymbol{r}_2^r + \boldsymbol{r}_3^r. \tag{1.1}$$

Differentiating, a velocity vector loop is generated

$$\boldsymbol{v}^d = \dot{\boldsymbol{r}}_1^r + \dot{\boldsymbol{r}}_2^r + \dot{\boldsymbol{r}}_3^r = \dot{\boldsymbol{r}}^d. \tag{1.2}$$

In component form, for the x components of position

$$r_x^d = r_1^r cos\Theta_1 + r_2^r cos\Theta_2 + r_3^r sin\Theta_3, \tag{1.3}$$

for the y components of position

$$r_y^d = r_1^r sin\Theta_1 + r_2^r sin\Theta_2 \tag{1.4}$$

and for the z components of position

$$r_z^d = r_3^r cos\Theta_3, \tag{1.5}$$

where the link lengths are given by $r_i^r = \boldsymbol{jjr}_i^r\boldsymbol{jj} = \sqrt{\boldsymbol{r}_i^r \quad \boldsymbol{r}_i^r}$ (for $i = 1, 2, 3$) and the planar (x-y) angles are measured counter-clockwise from horizontal right (Figure 1.1). The velocities can subsequently found from differentiating the component equations of Equation 1.3-1.5, yielding, for the x components of velocity

$$v_x^d = \quad r_1^r\dot{\Theta}_1 sin\Theta_1 \quad r_2^r\dot{\Theta}_2 sin\Theta_2 + r_3^r\dot{\Theta}_3 cos\Theta_3, \tag{1.6}$$

for the y components of velocity,

$$v_y^d = r_1^r\dot{\Theta}_1 cos\Theta_1 + r_2^r\dot{\Theta}_2 cos\Theta_2 \tag{1.7}$$

and for the z components of velocity

$$v_z^d = \quad r_3^r\dot{\Theta}_3 sin\Theta_3. \tag{1.8}$$

The total velocity of the droplet coming out of the dispenser is the velocity of the dispenser plus the relative droplet dispenser velocity ($\Delta\boldsymbol{v}^e$):

$$\boldsymbol{v} = \boldsymbol{v}^d + (\boldsymbol{v} \quad \boldsymbol{v}^d) = \boldsymbol{v}^d + \Delta\boldsymbol{v}^d. \tag{1.9}$$

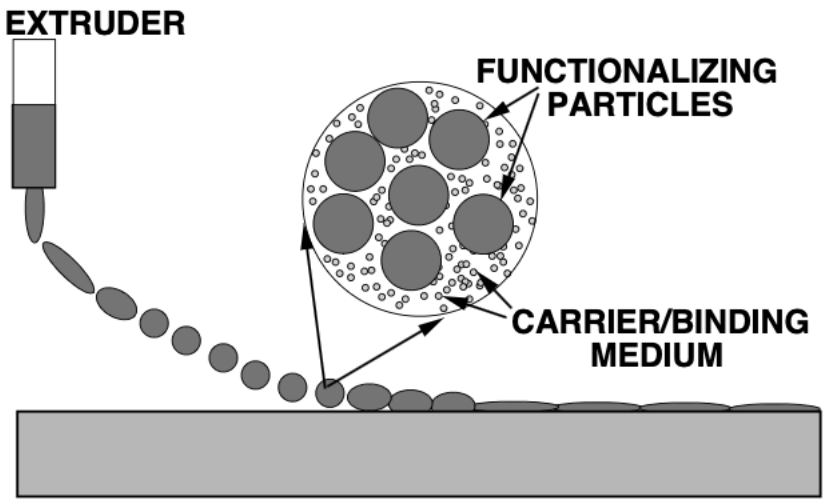

Figure 1.2: Deposition of "droplets" of a mixtures.

1.3 Dynamics of released material

The dispensed material, which is released as "syrup-like" droplets of multiphase mixtures as the robotic system moves (Figure 1.2), is represented by a series of lumped masses. Macro-particle interaction within the droplet mixture, as well as the interaction between the droplets of mixtures are not considered in this part of the analysis, since this is not relevant for the overall macroscale robot path planning calculations of the printing process. However, detailed modeling of the dynamics of the released material is discussed later in the chapter.

Accordingly, for the lumped mass analysis, at a snapshot in time, and an arbitrary i^{th} lumped mass in the system, whose dynamics are governed by

$$m_i \ddot{\mathbf{r}}_i = \mathbf{\Psi}_i^{elec} + \mathbf{\Psi}_i^{grav} + other\ external\ forces \mathbf{\Psi}_i^{tot}(\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{N_m}), \tag{1.10}$$

where $\mathbf{r}_i$ is the position vector of the i^{th} lumped mass, $\mathbf{\Psi}^{elec}$ are the externally applied electrical force, $\mathbf{\Psi}_i^{grav} = m_i \mathbf{g}$ is the gravitational force and $\mathbf{\Psi}_i^{tot}$ is the total of the forces acting on the i^{th} lumped mass.

Remarks: Often when a complex fluidized mixture exits a dispenser, a droplet is formed. However, it can initially exit as a long slender filament, that will break up into droplets, due to instabilities. The breakup of a long column of fluid from a dispenser, possessneting slight perturbations (longitudinal waviness) was first experimentally studied by Plateau in 1873, who determined that a vertically falling water stream into drops if its wavelength is greater than approximately 3.13-3.18 times its diameter. Subsequently, Rayleigh analytically proved that a wavy falling column of liquid (with circular cross-section) will break up into drops if its wavelength exceeded its circumference. This type of instability is driven by surface tension, which forces fluids to minimize their surface area. See Papageorgiou [13] and Eggers [14] for more details. In the case of a dispensed viscous medium, the degree of fluidity/viscosity influences surface tension. We refer the reader to Zohdi [15] for detailed simulation of this phenomena. In the present analysis, we assume that the droplet has been formed. Later in the chapter this stream/droplet phenomenon is discussed further.

1.4 Construction of targeted electrified field

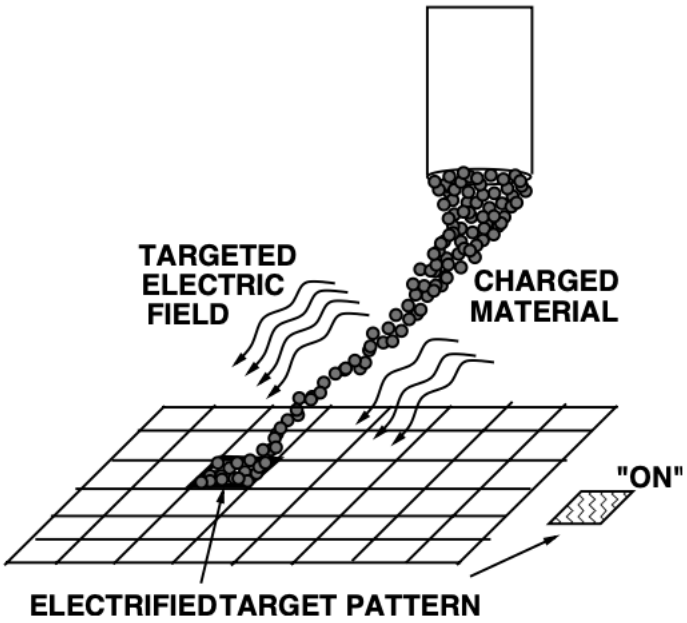

Figure 1.3: Highly targeted feature patterns that are electrified.

A key in digital printing is the targeted zonal electrification of the substrate to guide the material to the precise desired locations. This is done, as in the classical "xerox" process by charging pixels on the substrate, turning them on and off in a controlled manner, to attract the deposited material to key locations. To construct an electrified pattern, consider an electric field due to charged external point source, representing one pixel on the substrate (Figure 1.3), which is governed by Gauss' law:

$$\int_A \mathbf{D} \quad \mathbf{n}\, dA = \epsilon E 4\pi \mathbf{jj r} \quad \mathbf{r}_o \mathbf{jj}^2 = \int_V Q_c\, dV = q_p \;\;) \;\; \mathbf{E} = \frac{q_p}{4\pi\epsilon \mathbf{jj r} \quad \mathbf{r}_o \mathbf{jj}^2}\mathbf{n}, \tag{1.11}$$

where $\mathbf{D}$ is the electric field flux, $\mathbf{E}$ is the electric field ($E = \quad \mathbf{jjEjj}$), Q_c is the charge per unit volume, q_p is the total charge at the point source, $\mathbf{n}$ is the normal-outward unit vector and $\mathbf{jj r} \quad \mathbf{r}_o \mathbf{jj}$ is the distance from the point source. To create the electric field associate with the entire set of pixels, one simply superposes the individual fields. Thus, using the point sources, the process is:

1. **ACTIVATION OF POINT SOURCES:**

$$\mathbf{E}_p = \frac{q_p}{4\pi\epsilon \mathbf{jj r}_i \quad \mathbf{r}_p \mathbf{jj}^2}\mathbf{n}_{i\,!\,p}. \tag{1.12}$$

2. **CONSTRUCTION OF INDUCED E-FIELD:**

$$\mathbf{E}^{ext}(\mathbf{r}_i) = \sum_{p=1}^{N_c} \frac{q_p}{4\pi\epsilon \mathbf{jj r}_i \quad \mathbf{r}_p \mathbf{jj}^2}\mathbf{n}_{i\,!\,p}, \tag{1.13}$$

where N_c are the number of charged point sources. The induced electrical force on the released material is $\mathbf{\Psi}_i^{elec} = q_i \mathbf{E}^{ext}(\mathbf{r}_i)$, where q_i is the charge of released material and $\mathbf{E}^{ext}$ is the electrical field constructed by the points on the substrate.

Remark: A general electromagnetic field (as opposed to the previous purely electrical field) can be introduced in two parts: (1) Lorentz forces (for charged/electrified materials) and (2) magnetic forces (for magnetic materials). In mathematical form,

$$\mathbf{\Psi}_i^{e+m} = \mathbf{\Psi}_i^{lor,e+m} + \mathbf{\Psi}_i^{mag} = \underbrace{q_i(\mathbf{E}^{ext} + \mathbf{v}_i \quad \mathbf{B}^{ext})}_{\mathbf{\Psi}_i^{lor,e+m}} + \mathbf{\Psi}_i^{mag}, \tag{1.14}$$

where Lorentz-induced forces from independent external fields are $\mathbf{E}^{ext}$ and $\mathbf{B}^{ext}$ is $\mathbf{\Psi}_i^{lor,e+m}$. The terms $\mathbf{E}^{ext}$ and $\mathbf{B}^{ext}$ are considered to be externally controlled and uncoupled from one another (Jackson [17]). In addition to using electrical fields to guide material to the target location, electrical fields may be needed help extract the highly viscous particle-laden material complex from the dispenser. Although this issue is not simulated here, we refer the reader to Zohdi [16] for more details.

1.5 Drag forces

Because the process could be in an open air environment, drag effects can be relevant. For example, we will employ a general phenomenological model

$$\mathbf{\Psi}_i^{drag} = \frac{1}{2}\rho_a C_{Di} \mathbf{jj}\mathbf{v}^f \quad \mathbf{v}_i \mathbf{jj}(\mathbf{v}^f \quad \mathbf{v}_i) A_i^D, \tag{1.15}$$

where C_{D_i} is the drag coefficient, A_i^D is the reference area, which for a sphere is $A_i^D = \pi R_i^2$, ρ_a is the density of the ambient fluid environment and $\mathbf{v}^f$ is the velocity of the surrounding medium which, in the case of interest, is air which is included in the equation of motion for the i^{th} droplet in the system

$$m_i \dot{\mathbf{v}}_i = \mathbf{\Psi}_i^{tot} = \mathbf{\Psi}_i^{elec} + \mathbf{\Psi}_i^{grav} + \mathbf{\Psi}_i^{drag}. \tag{1.16}$$

The empirical drag coefficient varies with Reynolds number. For example, consider the following piecewise relation (Chow [18]):

- For $0 < Re \quad 1$, $C_{Di} = \frac{24}{Re}$,
- For $1 < Re \quad 400$, $C_{Di} = \frac{24}{Re^{0.646}}$,
- For $400 < Re \quad 3 \quad 10^5$, $C_{Di} = 0.5$,
- For $3 \quad 10^5 < Re \quad 2 \quad 10^6$, $C_{Di} = 0.000366 Re^{0.4275}$,
- For $2 \quad 10^6 < Re < \mathbf{1}$, $C_{Di} = 0.18$,

where the local Reynolds number for a drop is $Re \frac{2R\rho_a \mathbf{jj}\mathbf{v}^f \quad \mathbf{v}_i \mathbf{jj}}{\mu_f}$ and μ_f is the fluid viscosity.[1] We note that in the zero Reynolds number limit the drag is Stokesian. Using the piecewise relation reduces the drag at the lower Reynolds number regimes.

Remark: The piecewise drag law of Chow [18] is a mathematical description for the Reynolds number over a wide range and is a curve-fit of extensive data from Schlichting [98].

1.6 Heat transfer

For the thermodynamics of lumped masses/drop, we consider conduction, convection and radiation. Thus, for each lumped mass/drop ($i = 1, 2, ..., N_d$),

$$m_i C_i \dot{\theta}_i = \mathbf{Q}_i + \mathbf{H}_{ci} + \mathbf{H}_{ri} + \mathbf{H}_{di}, \tag{1.17}$$

where θ_i is the temperature in degrees Kelvin, C_i is the heat capacity per unit mass, $\mathbf{Q}_i$ represents the conductive contribution from the substrate, when in makes contact, the robotic dispenser and support

[1] The viscosity coe cient for air is $_f$ = 0:000018 Pa/s.

structures (if any), H_{ci} represents surrounding convection, H_{ri} represents the (infrared) radiation and H_{di} represents the drag heating. It is assumed that the temperature fields are uniform within the (small) elements surrounding the lumped masses. This assumption is justified, i.e. a lumped thermal model (associates with the lumped masses), ignoring temperature gradients and assuming a uniform temperature when the Biot number is small. The Biot number for a cylindrical element scales with the ratio of the element volume (V) to the element surface area (A^s), $\frac{V}{A^s} = \frac{4\pi R^3/3}{4\pi R^2} = \frac{R}{3}$ (R is the radius of the droplet), which indicates that a uniform temperature distribution is appropriate, since the elements, by definition, are small. Assuming that the fields are uniform in each element allows for the following (for the $i = 1, 2, ..., N_d$ lumped masses)

$$Q_i \quad {}_i \frac{\theta_{sub} \quad \theta_i}{jj\boldsymbol{r}_i \quad \boldsymbol{r}_{sub}jj} A_i^c, \tag{1.18}$$

where ${}_i$ is the conductivity, θ_{sub} is the temperature of the substrate, $\boldsymbol{r}_{sub}$ is the position of the substrate and $A_i^c = \pi R_i^2$ is the contact area associated with the lumped mass/substrate pair (ij). This yields, including standard convection and radiation terms

$$m_i C_i \frac{d\theta_i}{dt} = \underbrace{{}_i \frac{\theta_{sub} \quad \theta_i}{jj\boldsymbol{r}_i \quad \boldsymbol{r}_{sub}jj} A_i^c}_{Q_i(when\ in\ contact)} \quad \underbrace{h_i A_i^s(\theta_i \quad \theta_s)}_{H_{ci}} \quad \underbrace{\epsilon_i \beta A_i^s(\theta_i^4 \quad \theta_s^4)}_{H_{ri}}, \tag{1.19}$$

where h_i is the convection coefficient, $A^s \quad 4\pi R^2$, $0 \quad \epsilon_i \quad 1$ is the radiative efficiency and $\beta = 5.67 \quad 10^{\ 8} W/(m^2 K^4)$ is the Stefan-Boltzmann constant. Upon contact the drop is assumed to stick and collapse into a roughly cylindrical shape with the following areas: (a) substrate contact area=πR^2 (b) convective area=$\pi R^2 + 2\pi RL$, where L is determined by the equating the volume of material in the spherical state to the cylindrical configuration ($\pi R^2 L = 4\pi R^3/3$) that is exposed and (c) radiative area=$\pi R^2 + 2\pi RL$ (neglecting the radiative exchange with the substrate). Finally, although potentially small, for the drag-heating rate, we take the inner-product of the drag force with the relative velocity of the drop to the surrounding environment, and insert it into the First Law of Thermodynamics:

$$H_{di} = {}_i \boldsymbol{\Psi}_i^{drag} \quad (\boldsymbol{v}^f \quad \boldsymbol{v}_i) = {}_i \frac{1}{2} \, {}_a C_{Di} jj\boldsymbol{v}^f \quad \boldsymbol{v}_i jj(\boldsymbol{v}^f \quad \boldsymbol{v}_i) A_i^D \quad (\boldsymbol{v}^f \quad \boldsymbol{v}_i) = {}_i \frac{1}{2} \, {}_a C_{Di} jj\boldsymbol{v}^f \quad \boldsymbol{v}_i jj^3 A_i^D; \tag{1.20}$$

where $0 \quad \gamma_i \quad 1$ is the frictional heating efficiency. If one then includes convective and radiative cooling, this yields

$$m_i C_{i\,-i} = \underbrace{{}_i \frac{sub \quad i}{jj\boldsymbol{r}_i \quad \boldsymbol{r}_{sub}jj} A_i^c}_{Q_i(when\ in\ contact)} \quad \underbrace{h_i A_i^s(\ _i \quad \ _s)}_{H_{ci}} \quad \underbrace{{}_i \ A_i^s(\ _i^4 \quad \ _s^4)}_{H_{ri}} + \underbrace{\frac{1}{2} \, {}_a C_{Di} jj\boldsymbol{v}^f \quad \boldsymbol{v}_i jj^3 A_i^D}_{H_{di}}; \tag{1.21}$$

Remark: To quantify the convection coefficient (h), we consider the Nusselt number (Nu), which is the ratio between the heat transfer of convection to heat transfer of conduction

$$Nu \frac{hL}{\quad} \ ) \ h = \frac{Nu}{L}, \tag{1.22}$$

where $L = 2R$ is the length scale associated with the particle radius R and is the fluid conductivity. The Nusselt number is related to the Reynolds number

$$Re \frac{2\rho R jj\boldsymbol{v}^f \quad \boldsymbol{v}_i jj}{\mu}, \tag{1.23}$$

and Prandtl number

$$Pr \frac{c_p \mu}{\quad}, \tag{1.24}$$

by (Whitaker[20])

$$Nu \quad 2 + (0.4Re^{1/2} + 0.06Re^{2/3}) Pr^{0.4} (\frac{\mu}{\mu_s})^{0.25}, \tag{1.25}$$

where μ is the surrounding fluid viscosity at the surrounding and μ_s is the viscosity of the fluid at the surface temperature. Thus, once one has the Nusselt number, one can post-process the convection coefficient.[2]

1.7 Effective properties

In the analysis to follow, the properties of a droplet, which is a mixture of microheterogeneous materials, will be determined from effective medium theories.

1.7.1 Effective thermal conductivity

Over the last century, estimates for micro-heterogeneous materials, of which porous materials is a subclass, have been developed. A particularly useful set estimates, in fact rigorous bounds are those developed in the early 1960's by Hashin and Shtrikman[21-23]. For a two phase material, comprised of isotropic phases with an overall isotropic response, they specifically read as

$$\underbrace{{}_1 + \frac{\nu_2}{\frac{1}{{}_2 - {}_1} + \frac{1-\nu_2}{3{}_1}}}_{,} \qquad \underbrace{{}_2 + \frac{1-\nu_2}{\frac{1}{{}_1 - {}_2} + \frac{\nu_2}{3{}_2}}}_{,^+}, \tag{1.26}$$

where the conductivity of phase 2 (with volume fraction ν_2) is larger than phase 1 ($_2 \quad _1$). For the purposes of this present analysis ν_2 corresponds to the solid particle material and phase 1 will be considered as the binding "material". The volume fractions have to sum to unity:

$$\nu_1 + \nu_2 = 1 \;\;) \;\; \nu_1 = 1 - \nu_2. \tag{1.27}$$

The Hashin-Shtrikman bounds are the tightest bounds known when the only information is the volume fraction of the particles-in other words, no other structural or statistical information is known.

1.7.2 Effective density of the material

The effective density of a mixture for a two-phase materials can directly be determined by

$$\frac{m}{V}\rho \;\; \hbar\rho(\boldsymbol{x})i_V \frac{1}{V}\int_V \rho(\boldsymbol{x})\, dV = \frac{1}{V}\left(\int_{V_1} \rho_1\, dV + \int_{V_2} \rho_2\, dV\right) = \nu_1\rho_1 + \nu_2\rho_2. \tag{1.28}$$

Furthermore, the heat capacity is approximated by

$$\begin{aligned} C \;\; \hbar C(\boldsymbol{x})i_V \qquad & \frac{1}{V}\int_V C(\boldsymbol{x})\, dV \\ = \;\; & \frac{1}{V}\left(\int_{V_1} C_1\, dV + \int_{V_2} C_2\, dV\right) \\ = \;\; & \nu_1 C_1 + \nu_2 C_2 \end{aligned} \tag{1.29}$$

The density-oriented calculations are not estimates *they are exact.* The effective density of a mixture for a two-phase materials can directly be determined by

$$m = \rho \;\; V = (\nu_1\rho_1 + \nu_2\rho_2)\frac{4}{3}\pi R^3, \tag{1.30}$$

where $V = \frac{4}{3}\pi R^3$, while the effective thermal mass is

$$mC = \rho \;\; C \;\; V = (\nu_1\rho_1 + \nu_2\rho_2)(\nu_1 C_1 + \nu_2 C_2)\frac{4}{3}\pi R^3. \tag{1.31}$$

[2] In the analysis to follow, we assume $_s$.

1.7.3 Effective charge-carrying capacity of the material

$$q\ \langle q(\boldsymbol{x})\rangle_V \frac{1}{V}\int_V q(\boldsymbol{x})\,dV\ =\ \frac{1}{V}\left(\int_{V_1} q_1\,dV\ +\ \int_{V_2} q_2\,dV\right)\ =\ {}_1\langle q_1\rangle_{V_1}\ +\ {}_2\langle q_2\rangle_{V_2}: \tag{1.32}$$

Depending on the materials under consideration, either the carrier material and/or the particles could carry a charge.

Remark: There exist a large number of effective property estimation techniques, and we refer the reader to Torquato [24], Jikov et al. [25], Hashin [23], Mura [26], Markov [27] for theoretical aspects and for more computationally-oriented approaches, Ghosh [28], Ghosh and Dimiduk [29], Matous et al [30] and Zohdi and Wriggers [31], Zohdi [32]. It is important to note that direct numerical approaches, discretizing the entire particle-laden heterogeneous domain with a very fine mesh, are quite accurate, but inordinately computationally expensive. If is for this reason that multiscale models are frequently used, which employ less computationally-intensive effective property models in most of the domain and detailed microstructural models in critical regions (see Zohdi and Wriggers [31] for reviews).

1.8 Integration of the equations

With the governing equations established, one can then integrate Equation 1.10 to obtain the velocity for the i^{th} lumped mass with a simple Forward-Euler integration[3]

$$\boldsymbol{v}_i(t+\Delta t)\ \ =\ \ \boldsymbol{v}_i(t)+\frac{1}{m_i}\int_t^{t+\ t}\boldsymbol{\Psi}_i^{tot}(t)\,dt\ \ \ \boldsymbol{v}_i(t)+\frac{\Delta t}{m_i}\boldsymbol{\Psi}_i^{tot}(t), \tag{1.33}$$

and the position for the i^{th} lumped mass by applying the integration process again:

$$\boldsymbol{r}_i(t+\Delta t)\ \ \ \boldsymbol{r}_i(t)+\Delta t\boldsymbol{v}_i(t), \tag{1.34}$$

which provides an updating scheme for the set equations for $i = 1, 2, 3, ..., N_d$ lumped masses. For the thermal states

$$\begin{aligned}\theta_i(t+\Delta t)\ \ =\ \ \theta_i(t)+\frac{1}{m_iC_i}\left(\int_t^{t+\ t}Q_i\,dt+\int_t^{t+\ t}H_{ci}\,dt+\int_t^{t+\ t}H_{ri}\,dt+\int_t^{t+\ t}H_{di}\,dt\right)\\ \theta_i(t)+\frac{\Delta t}{m_iC_i}\left(Q_i(t)+H_{ci}(t)+H_{ri}(t)+H_{di}(t)\right).\end{aligned} \tag{1.35}$$

1.8.1 Overall solution algorithm

The algorithm is as follows:

(1.36)

```
(1) COMPUTE ROBOT SYSTEM POSITION TIME = t :

(2) SET i = 1 :

(3) IF i > N_d THEN GO TO (5)

(4) IF i   N_d THEN : (FOR LUMPED MASS i)

    (a) COMPUTE POSITION : r_i(t +  t)
    (b) COMPUTE TEMPERATURE : _i(t +  t)
    (c) GO TO (3) AND NEXT DROP=LUMPED MASS (i = i + 1)

(5)INCREMENT TIME : t = t +  t
```

[3] More sophisticated implicit solution methods for strongly interacting multi-component systems can be found in Zohdi [33-44].

Parameter	Value
Link one angular velocity	ω_1 = 10 rad/s
Link two angular velocity	ω_2 = 1 rad/s
Link three angular velocity	ω_3 = 10 rad/s
Dispensing velocity	$v^{d\ e}$ = 1 m/s
Link one	L_1 = 1m
Link two	L_2 = 1m
Link three	L_3 = 0.5m
Surrounding cooling fluid velocity	$\mathbf{v}_f$ = (10; 0; 0) m/s
Surrounding convective cooling coefficient	h = 100 $W/(m^2K)$
Volume fraction of phases 2	v_1 = 0.75 and v_2 = 0.25)
Phase 1 mass density	ρ_1 = 2000 kg/m^3
Phase 2 mass density	ρ_2 = 7000 kg/m^3
Temperature of the robot	400 K
Phase 1 heat capacity of the droplet	C=1000 J/(kg-K)
Phase 2 heat capacity of the droplet	C=1000 J/(kg-K)
Ambient temperature	θ_o = 325 K
Droplet radius	R = 0.001 m
Phase 1 thermal conductivity	κ_1 = 10W/(m-K)
Phase 2 thermal conductivity	κ_2 = 20W/(m-K)
Pixel Charge	q_p = (1 10 ^{7}; 0) C
Pixel Grid	10 10 pixels
Per unit volume charge for Phase 1	$\hat{q}_1$ = 0 C/m^3
Per unit volume charge for Phase 2	$\hat{q}_2$ = 1000 C/m^3
Radiative efficiency	= 0.75

Table 2.1: Material parameters used in the example.

2 Example

Consider the three-link mechanism introduced at the beginning of this chapter, with material dispensed from the printer. The system parameters are shown in Table 2.1. The material that exits the printer initially has the temperature and velocity of the print head. The angular velocities are constant in this simple example. Ten-thousand time steps were used. The wall-clock simulation time was on the order of 0.03 seconds on a Macbook-Pro using a FORTRAN-90 code written by the author. The results are shown in Figures 2.4-2.6, with and without the electrical charging. Clearly, with the charge, one can print within the target, despite the inaccuracy of the robot. The speed at which this type of simulation can be completed allows one to answer the inverse question of what the combination of parameters should be for a desired result. In order to cast the objective mathematically, we set the problem up as a Machine Learning Algorithm (MLA). Following Zohdi [46], we formulate the objective as a cost-function minimization problem whereby a pattern at time $t = T = 1$ second is given by (Figure 2.1)

- Specifying the desired locations of the drops: $\mathbf{r}_i^{des}(t)$, $i = 1, ..., N_d$ at time $t = T$,
- Determining where the drops have been placed by a trial set of parameter: $\mathbf{r}_i^{gen}(t)$, $i = 1, ..., N_d$ at time $t = T$,
- Determining the difference the desired and generated patterns:

$$\Pi = \frac{\sum_{i=1}^{N_d} \| \mathbf{r}_i^{des} - \mathbf{r}_i^{gen} \|}{\sum_{i=1}^{N_d} \| \mathbf{r}_i^{des} \|}, \tag{2.1}$$

- Systematically minimize Equation 2.1, $min \Pi$, by varying the design parameters: $\{ {}^{i}f_{1}^{i}, {}^{i}_{2}, {}^{i}_{3}, {}^{i}_{4}, ..., {}^{i}_{N} \} \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \Delta v^d, q_p$.
- The system parameter search is conducted within the constrained ranges of $\dot{\Theta}_1^{(-)} \le \dot{\Theta}_1 \le \dot{\Theta}_1^{(+)}$, $\dot{\Theta}_2^{(-)} \le \dot{\Theta}_2 \le \dot{\Theta}_2^{(+)}$, $\dot{\Theta}_3^{(-)} \le \dot{\Theta}_3 \le \dot{\Theta}_3^{(+)}$, $\Delta v^{e(-)} \le \Delta v^d \le \Delta v^{e(+)}$ and $q_p^{(-)} \le q_p \le q_p^{(+)}$. These upper and lower limits would, in general, be dictated by what is physically possible with the machinery and materials available.

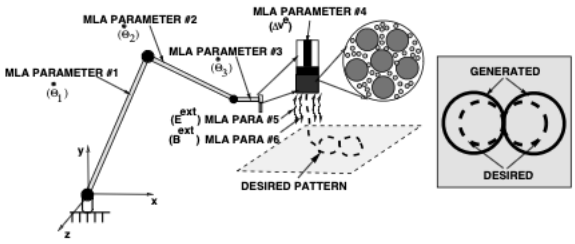

Figure 2.1: Machine Learning Algorithm model problem of a robot with a dispensing printer head.

2.1 System parameter search: Machine Learning Algorithm (MLA)

Here we follow Zohdi [45, 46] in order to minimize Equation 2.1, which we will refer to as a "cost function". Cost functions such as Equation 2.1 are nonconvex in design parameter space and often nonsmooth. Their minimization is usually difficult with direct application of gradient methods. This motivates nonderivative search methods, for example those found in Machine Learning Algorithms (MLA's). One of the most basic subset of MLA's are so-called Genetic Algorithms (GA's). Typically, one will use a GA first in order to isolate multiple local minima, and then use a gradient based algorithm in these locally convex regions or reset the GA to concentrate its search over these more constrained regions. GA's are typically the simplest schemeto start the analysis, and one can, of course, use more sophisticated methods if warranted. For a review of GA's, see the pioneering work of John Holland (Holland [47]), as well as Goldberg [48], Davis [49], Onwubiko [50], Lagaros et al. [51], Papadrakakis et al. [52-55] and Goldberg and Deb [56]. The GA approach is extremely well-suited for nonconvex, nonsmooth, multicomponent, multistage systems, and involves the following essential concepts:

1. **POPULATION GENERATION:** Generate system population: ${}^{i}f^{i}_{1},{}^{i}_{2},{}^{i}_{3},{}^{i}_{4},...,{}^{i}_{N}\,g = f\dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \Delta v^d, q_p g^i$
2. **PERFORMANCE EVALUATION:** Compute fitness/performance of each genetic string: $\Pi(^i)$ and rank them $(i = 1, ..., N)$
3. **MATING PROCESS:** Mate pairs/produce offspring: ${}^{i}\Phi^{(I)i} + (1 \quad \Phi^{(I)})^{i+1}$ where $0 \quad \Phi \quad 1$ (Figure 2.2
4. **GENE ELIMINATION:** Eliminate poorly performing genetic strings, keep top parents and generated offspring
5. **POPULATION REGENERATION:** Repeat the process with the new gene pool and new *random* genetic strings
6. **SOLUTION POST-PROCESSING:** Employ gradient-based methods afterwards in the local "valleys"- *if smooth enough*

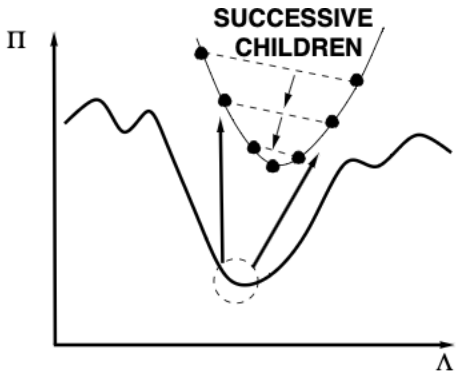

Figure 2.2: The basic action of a genetic algorithm.

DESIGN	$\dot{\Theta}_1$	$\dot{\Theta}_2$	$\dot{\Theta}_3$	Δv^d	q_p	Π
1	-5.94	-0.79	12.20	1.13	0.0000001285	0.0077391595
2	-8.48	-0.62	12.23	1.59	0.0000001169	0.0077464996
3	-9.13	-0.61	12.62	1.28	0.0000001214	0.0077469398
4	-7.91	-0.77	8.47	1.57	0.0000001237	0.0077470045
5	-9.06	-0.54	8.12	1.36	0.0000001122	0.0371113337
6	-8.47	-0.75	7.97	1.53	0.0000001310	0.0403717650
7	-8.50	-0.81	7.78	1.27	0.0000001207	0.0414568277
8	-9.00	-0.55	10.80	1.46	0.0000001462	0.0455584033
9	-8.37	-0.75	7.82	1.24	0.0000001463	0.0462887025
10	-8.74	-0.68	10.53	1.28	0.0000001227	0.0560576224

Table 2.2: The top 10 system parameter performers.

2.1.1 Algorithmic specifics

Following Zohdi [45, 46], the algorithm is as follows:

- **STEP 1:** Randomly generate a population of S starting genetic strings, $^i, (i = 1, 2, 3, ..., S)$: $^i f^i_{1,} {}^i_2 {}^i_3 {}^i_4, ..., {}^i_N g f \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \Delta v^d, q_p g$
- **STEP 2:** Compute fitness of each string $\Pi(^i)$, (i=1, ..., S)
- **STEP 3:** Rank genetic strings: ^{i}, (i=1, ..., S)
- **STEP 4:** Mate nearest pairs and produce two offspring, (i=1, ..., S)
  $^i\Phi^{(I)i} + (1 \quad \Phi^{(I)})^{i+1}, \qquad ^{i+1}\Phi^{(II)i} + (1 \quad \Phi^{(II)})^{i+1}$
- **STEP 5:** Eliminate the bottom $M < S$ strings and keep top $K < N$ parents and top K offspring (K offspring+K parents+M=S)
- **STEP 6:** Repeat STEPS 1-6 with top gene pool (K offspring and K parents), plus M new, randomly generated, strings
- **NOTE:** $\Phi^{(I)}$ and $\Phi^{(II)}$ are random numbers, such that 0 $\Phi^{(I)}$ 1, 0 $\Phi^{(II)}$ 1, which are different for each component of each genetic string
- **OPTION:** Rescale and restart search around best performing parameter set every few generations

Remark 1: If one selects the mating parameter Φ to be greater that one and/or less than zero, one can induce "mutations". i.e. characteristics that neither parent possesses. However, this is somewhat redundant with introduction of new random members of the population in the current algorithm.

Remark 2: If one does not retain the parents in the algorithm above, in is possible that inferior performing offspring may replace superior parents. Thus, top parents should be kept for the next generation. This guarantees a monotone reduction in the cost function. Furthermore, retained parents do not need to be re-evaluated-making the algorithm less computationally less expensive, since these parameter sets do not have to be reevaluated (or ranked) in the next generation. Numerous studies of the author have shown that advantages parent retention outweighs inbreeding, for sufficiently large population sizes. Finally, we remark that this algorithm is easily parallelizable.

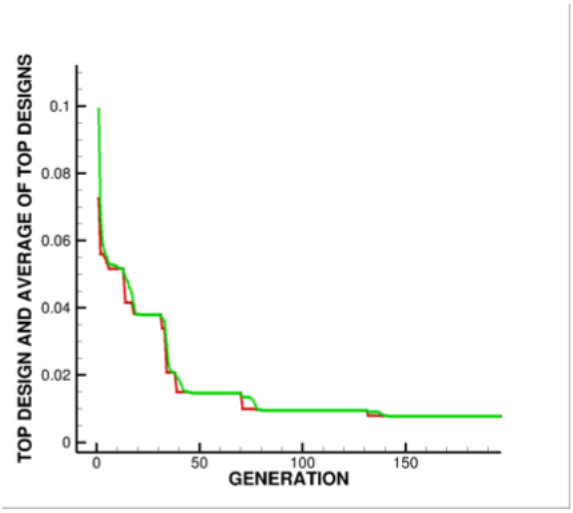

Figure 2.3: LEFT-USING THE Objective FUNCTION (II). Machine learning output, generation after generation. Shown are the best performing gene (design parameter set, in **red**) as a function of successive generations, as well as the average performance of the population of the top four genes (designs, in **green**).

2.2 MLA electrodynamic example

As a model problem, we used the result of the previous numerical example that generated the results and pattern in Figure 2.4 using

- $\dot{\Theta}_1 =$ 10 rad/s,
- $\dot{\Theta}_2 =$ 1 rad/s,
- $\dot{\Theta}_3 = 10$ rad/s,
- $\Delta v^d = 1$ m/s and
- $q_p = 10^{\ 7}$ C.

We used the following MLA settings:

- Number of design variables: 5,
- Search domain for $\dot{\Theta}_1$: $\dot{\Theta}_1 = \ 5 \quad \dot{\Theta}_1 \quad 30 = \dot{\Theta}_1^+$,
- Search domain for $\dot{\Theta}_2$: $\dot{\Theta}_2 = \ 0.5 \quad \dot{\Theta}_2 \quad 3 = \dot{\Theta}_2^+$,
- Search domain for $\dot{\Theta}_3$: $\dot{\Theta}_3 = 5 \quad \dot{\Theta}_3 \quad 30 = \dot{\Theta}_3^+$,
- Search domain for Δv^d: $\Delta v^{e,} = 0.5 \quad \Delta v^d \quad 3.5 = \Delta v^{e,+}$,
- Search domain for q_p: $q_p = 0 \quad q_p \quad 3 \quad 10^{\ 7} = q_p^+$,
- Population size per generation: 20,
- Number of parents to keep in each generation: 4,
- Number of children created in each generation: 4,
- Number of completely new genes created in each generation: 12 and
- Number of generations: 200.

The algorithm was automatically reset every *30 generations*. The entire 200 generation simulation, with 20 genes per evaluation (8000 total designs) took on the order of 5 minutes of a laptop, *making it ideal as a design tool.* Figure 2.3 (average top four genes performance and top gene performance) and Table 2.2 (values of the gene components) illustrate the results. The MLA/GA is able to home in of a variety of possible designs, including the one corresponding to the original set of parameters that generated the test pattern and alternatives that achieve virtually the same results. This allows system designers to more flexibility in parameter selection. We note that, for a given set of parameters, a complete simulation takes on the order of 0.03 seconds, thus over 100,000 parameter sets can be evaluated in an hour, *without even exploiting the inherent parallelism of the MLA.* For more detailed micromechanical information on the behavior of the optimal result, Discrete Element Methods (DEM) are needed, and are discussed in the summary of this chapter.

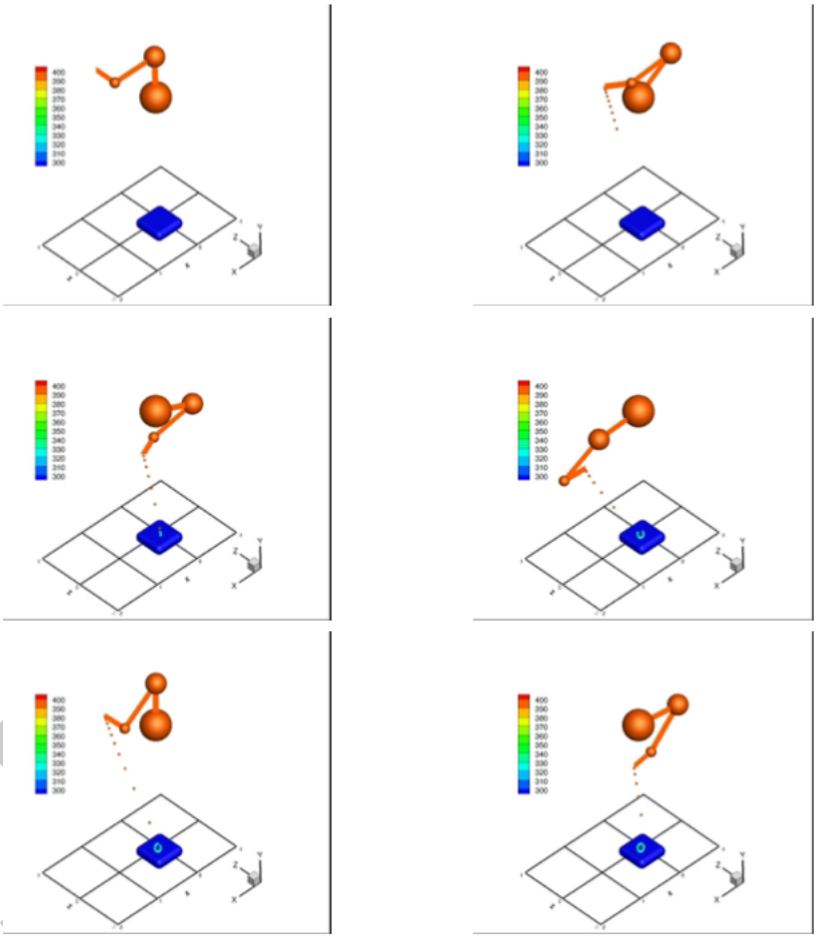

Figure 2.4: **WITH FAULT-TOLERANCE CHARGING**-A deposition pattern: left to right and top to bottom. The colors indicate the temperature. The robot temperature was 400 K (color coded orange) while the substrate was set as 300 K (color coded dark blue). The total time was $T = 1$ second and the time slabs are $T = 0, 1/6, 1/3, 1/2, 2/3$ and 1 seconds. The robot started at position $\Theta_1 = \pi/2$, $\Theta_2 = 0$ and $\Theta_3 = 0$.

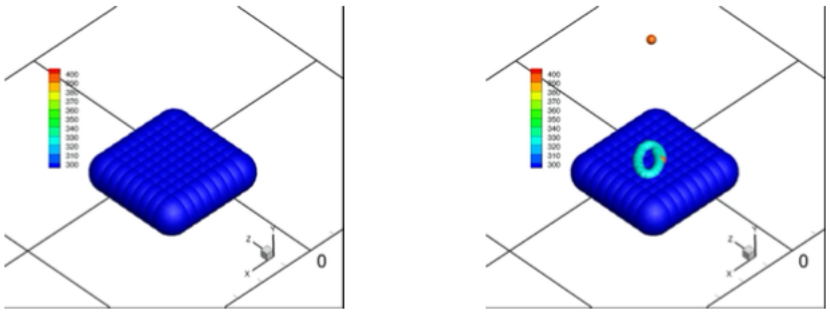

Figure 2.5: A zoom on the drops being deposited on the electrified pattern in Figure 2.4.

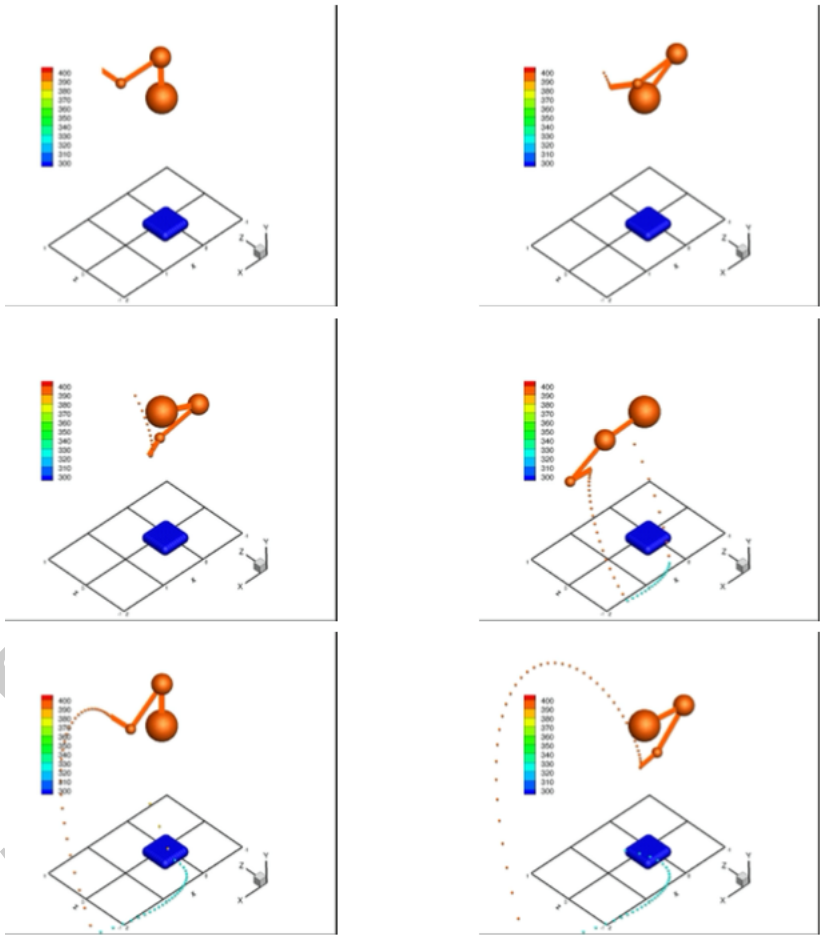

Figure 2.6: **WITHOUT FAULT-TOLERANCE CHARGING:** A deposition pattern: left to right and top to bottom. The colors indicate the temperature. The robot temperature was 400 K (color coded orange) while the substrate was set as 300 K (color coded dark blue). The total time was $T = 1$ second and the time slabs are $T = 0, 1/6, 1/3, 1/2, 2/3$ and 1 seconds. The robot started at position $\Theta_1 = \pi/2$, $\Theta_2 = 0$ and $\Theta_3 = 0$.

2.3 Extensions

In addition to the obvious use for industrial additive manufacturing processes, application of the methodology to the 3D bioprinting of flowing media containing cells is quite relevant. Over the last decade, there has been a steady increase in so-called bioprinting technology, whereby cells are combined with fluids containing nutrients, growth factors, etc, to make what is called a bioink. The slurry of material is then loaded into a 3D bioprinter that deposits the mixture, layer by layer onto a surface to build tissue-like structures, ligaments, cartilage, organs, for example on scaffolds or to make scaffolds. The liquid that the cells are immersed into provide the nutrients for the cells to remain alive. In some cases, the cells are encapsulated in a spherical shell within the fluid in order to keep the cells healthy. The field is relatively wide now (see [57-67]), but still has a common theme: the attempted deposition of a particle-embedded slurry. Some emerging variants of this technology have attempted to utilize electromagnetic fields. For example, magnetic 3D bioprinting attaches biocompatible magnetic nanoparticles to cells in order to magnetize them. Thereafter, the cells can be precisely and rapidly deposited onto patterns following the electromagnetic fields (see [68-76]). In this case, only a magnetic field is needed, since the cells are rendered magnetically sensitive due to added surfactants. We note that intrinsic magnetic (non-electromagnetic) forces can be approximated by a simple model, $\boldsymbol{\Psi}^{mag} = r\,(\boldsymbol{m}\quad \boldsymbol{B}^{ext}) = r\,(\gamma \boldsymbol{B}^{ext}\quad \boldsymbol{B}^{ext})$ (independently of the Lorentz forces), where γ is a material parameter that is related to the magnetization ($\boldsymbol{m}$; magnetic dipole properties, susceptibility, permeability, moment density, etc.) of the element (see Feynman et al. [77], Cullity and Graham [78], Boyer [79] or Jackson [17]). Thus, for a spatially varying magnetic field we have

$$\boldsymbol{\Psi}^{mag} = r\,(\gamma \boldsymbol{B}^{ext}\quad \boldsymbol{B}^{ext}) = \gamma\, r\,(\boldsymbol{B}^{ext}\quad \boldsymbol{B}^{ext}) = 2\gamma(r\,\boldsymbol{B}^{ext})\quad \boldsymbol{B}^{ext}, \tag{2.2}$$

or explicitly

$$\boldsymbol{\Psi}^{mag} = 2\ (r\,\boldsymbol{B}^{ext})\quad \boldsymbol{B}^{ext} = 2 \begin{bmatrix} \frac{\partial B_1}{\partial x_1} & \frac{\partial B_1}{\partial x_2} & \frac{\partial B_1}{\partial x_3} \\ \frac{\partial B_2}{\partial x_1} & \frac{\partial B_2}{\partial x_2} & \frac{\partial B_2}{\partial x_3} \\ \frac{\partial B_3}{\partial x_1} & \frac{\partial B_3}{\partial x_2} & \frac{\partial B_3}{\partial x_3} \end{bmatrix}^{ext} \begin{Bmatrix} B_1 \\ B_2 \\ B_3 \end{Bmatrix}^{ext}; \tag{2.3}$$

In this case, we could construct the following additional genetic string

$$^{mag} = f\frac{@B_1}{@x_1}; \frac{@B_1}{@x_2}; \frac{@B_1}{@x_3}; \frac{@B_2}{@x_1}; \frac{@B_2}{@x_2}; \frac{@B_2}{@x_3}; \frac{@B_3}{@x_1}; \frac{@B_3}{@x_2}; \frac{@B_3}{@x_3}g; \tag{2.4}$$

which would generalize the one used previously to read

$$= f\dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \Delta \boldsymbol{v}^d, \boldsymbol{E}^{ext}, \boldsymbol{B}^{ext},\ r\,\boldsymbol{B}^{ext}g, \tag{2.5}$$

explicitly,

$$= f\,\_1,\ \_2,\ \_3,\quad \boldsymbol{v}^d, E_1^{ext}, E_2^{ext}, E_3^{ext}, B_1^{ext}, B_2^{ext}, B_3^{ext}, \frac{\partial B_1}{\partial x_1}, \frac{\partial B_1}{\partial x_2}, \frac{\partial B_1}{\partial x_3}, \frac{\partial B_2}{\partial x_1}, \frac{\partial B_2}{\partial x_2}, \frac{\partial B_2}{\partial x_3}, \frac{\partial B_3}{\partial x_1}, \frac{\partial B_3}{\partial x_2}, \frac{\partial B_3}{\partial x_3}g. \tag{2.6}$$

3 Assignment

MODELING AND SIMULATION OF ROBOTIC 3D PRINTERS (100 POINTS)

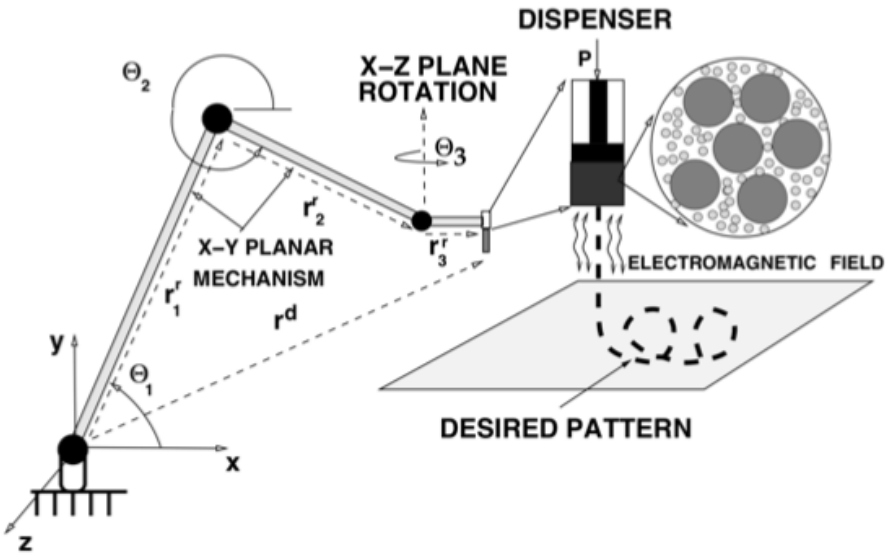

Figure 3.1: Coordinate System and Robot Schematic

Overview

In this project, you will simulate the operation of a free-form robotic 3D printer. Referring to Figure 1, the printer arm deposits electrically charged droplets above a substrate, with the substrate electrically charged in certain areas to attract the droplets. Using the simulation, you will employ a genetic algorithm to find control parameters that generate a printed pattern as close to a desired pattern as possible.

Problem Description

Part 1

The overall goal is to model the motion of the droplets. First, the robot arm dynamics can be modeled in order to obtain the starting position of the particles as they are released from the extruder located at the tip of the arm. The robot arm is modeled as a three-rod linkage system that has two rods rotating in the x-y plane and one rotating in x-z plane. Given the rod lengths $L_1; L_2; L_3$ and rod angular velocities $\omega_1; \omega_2; \omega_3$, the dispenser position can be found using the equations below:

$$x_d = L_1 \cos \theta_1 + L_2 \cos \theta_2 + L_3 \sin \theta_3 \tag{3.1}$$

$$y_d = L_1 \sin \theta_1 + L_2 \sin \theta_2 \tag{3.2}$$

$$z_d = L_3 \cos \theta_3 \tag{3.3}$$

where the rod angles are changing with respect to the constant rod velocity components and initial rod angles:

$$\theta_j = \theta_j(t) = \theta_j^0 + \omega_j t \qquad j = 1; 2; 3 \tag{3.4}$$

Dispenser velocity components can be found using the dispenser positions:

$$\underline{x}_d = \quad L_{1-1}\sin \quad _1 \quad L_{2-2}\sin \quad _2 + L_{3-3}\cos \quad _3 \tag{3.5}$$

$$\underline{y}_d = L_{1-1}\cos \quad _1 + L_{2-2}\cos \quad _2 \tag{3.6}$$

$$\underline{z}_d = \quad L_{3-3}\sin \quad _3 \tag{3.7}$$

When the droplet is generated in the simulation at each time step, it starts at the dispenser position $\boldsymbol{r}_i^0 = \boldsymbol{r}^d = r_0 + (x_d; y_d; z_d)$ and has the velocity $\boldsymbol{v}_i^0 = \boldsymbol{v}^d + \quad \boldsymbol{v}^d$ where $\quad \boldsymbol{v}^d$ is the extrusion speed of the dispenser. Now, the droplets can be initialized and therefore the droplet dynamics need to be de ned.

In Part 1, the de ned forces on the droplets are electric forces, drag forces, and gravitational forces. The gravitational force $\boldsymbol{F}_i^{grav}$ is de ned as:

$$\boldsymbol{F}_i^{grav} = (g_x; g_y; g_z) = (0; \quad m_i g; 0) \tag{3.8}$$

where m_i is the mass of a droplet and $g = 9{:}81 m{=}s^2$.

The electric force F_i^{elec} is de ned as:

$$F_i^{elec} = \sum_{p=1}^{N_c} \frac{q_p q_i}{4 \quad k r_i \quad r_p k^3}(r_i \quad r_p) \tag{3.9}$$

where q_p and r_p are the xed point charges and the respective locations, is the electric permittivity (equal to permittivity of free space $_0$), q_i and r_i are the droplet charges and locations, and N_c is the number of xed point charges. The xed-point charges are laid onto the printbed in a grid-fashion. The electric forces between droplets are neglected in this model.

The drag force $\boldsymbol{F}_i^{drag}$ is de ned as:

$$\boldsymbol{F}_i^{drag} = \frac{1}{2} \; _a C_{Di} k \boldsymbol{v}^f \quad \boldsymbol{v}_i k (\boldsymbol{v}^f \quad \boldsymbol{v}_i) A_i^D \tag{3.10}$$

where $_a$ is the medium density, v^f is the medium velocity, v_i is the droplet velocity, A_i^D is the drag reference area (equal to $A_i^D = \quad R^2$), and C_{Di} is the drag coe cient. C_{Di} depends on the Reynolds number of the droplet, which is de ned as:

$$Re = \frac{2R \; _a k v^f \quad v_i k}{_f} \tag{3.11}$$

where $_f$ is the medium viscosity and R is the droplet radius. The drag coe cient C_{Di} as a function of Reynolds number is as follows:

$$C_{Di} = \begin{cases} \frac{24}{Re}; & 0 < Re \quad 1 \\ \frac{24}{Re^{0.646}}; & 1 < Re \quad 400 \\ 0{:}5; & 400 < Re \quad 3 \quad 10^5 \\ 0{:}000366 Re^{0.4275}; & 3 \quad 10^5 < Re \quad 2 \quad 10^6 \\ 0{:}18; & Re > 2 \quad 10^6 \end{cases} \tag{3.12}$$

The previously mentioned density and charge values for the droplets are functions of the materials that make up the mixture. Therefore, the e ective density and e ective charge q for the droplets can be calculated using the volume fractions of the materials that compose the droplet mixture.

$$= (1 \quad v_2) \; _1 + v_2 \; _2 \tag{3.13}$$

$$q = (1 \quad v_2) \, hq_1 i + v_2 \, hq_2 i \tag{3.14}$$

where $_1$; $_2$ are the material density values and $hq_1 i \, ; hq_2 i$ are material charge capacity values.

These values will also be used to calculate the lumped droplet charge $q_i = V_i q$ and droplet mass $m_i = V_i$.

The forces on the droplets then can be combined to obtain the governing equation for a given i-th droplet:

$$m_i \ddot{r}_i = \quad {}_i^{tot} = F_i^{grav} + F_i^{elec} + F_i^{drag} \tag{3.15}$$

To obtain the time update equation for our model, velocity terms $v(t + \quad t)$ and $v(t)$ and position terms $r(t + \quad t)$ and $r(t)$ can be expanded using Taylor series about t to obtain the Forward Euler scheme:

$$r_i(t + \quad t) = r_i(t) + \quad t v_i(t) \tag{3.16}$$

$$v_i(t + \quad t) = v_i(t) + \quad t \dot{v}_i(t) \tag{3.17}$$

The main reason Forward Euler is used is because of its explicit nature, meaning that it requires no iteration to convergence the solution at a given time step. The position/velocity of the particle can be calculated at the next time step directly. However, it is conditionally stable (depending on time step size) and needs smaller time steps for accuracy. Time-stepping is only used in the rst part of the project. The droplets that reach the printbed are removed from the time-stepping algorithm and are permanently placed at their last calculated location.

The model is simulated for 3 seconds and the dispenser is extruding a droplet per time step, where the time-step size is $t = 0.001s$. After all the droplets exit the dispenser, the model is run for an extra time to let all droplets settle on the printbed.

Parameters used to model the system are given in the variable glossary.

Part 2

In this part of the project, the simpli ed free-fall model used enables us to analytically compute the droplet nal locations, using $m\ddot{r} = F^{grav}$. The electric and drag forces are removed to simplify the algorithm for the optimization process. Now, the inverse design problem is considered where a desired droplet pattern is provided and the required robot design parameters are requested. Instead of a gradient-based approach, a genetic algorithm is used to randomly generate design strings and use natural selection to nd the best- t string that produces a pattern closest to the given one.

A genetic algorithm is chosen over a gradient-based method as a way of avoiding calculating derivatives for a Newton-Raphson type of optimization. Also, gradient-based methods' inability to work with non-smooth and non-convex systems is avoided.

For the modeled system, the input parameters are the robot arm angular velocity components and the extrusion speed, which can be written as a design string:

$$\mathbf{\Lambda}^i = \left\{ {}_1; {}_2; {}_3; \quad \boldsymbol{v}^d \right\} \tag{3.18}$$

For each design string, the model can be run to calculate the droplet nal positions. The generated positions are then compared to the desired positions using a cost function :

$$= \frac{\sum_{i=1}^{N_d} \left\| \boldsymbol{r}_i^{des} \quad \boldsymbol{r}_i^{gen} \right\|}{\sum_{i=1}^{N_d} \left\| \boldsymbol{r}_i^{des} \right\|} \tag{3.19}$$

For applying a genetic algorithm, the algorithm is as follows:

Step 1: Generate S random genetic strings, where $_i \in [\ _i; \ {}_i^+]$

$$= (\ {}^{(1)}; \ {}^{(2)}; \ldots; \ {}^{(i)}; \ldots; \ {}^{(S)}$$

where

$${}^{(i)} = \begin{pmatrix} {}_1 & {}_1^{(i)} & {}_1^+ \\ {}_2 & {}_2^{(i)} & {}_2^+ \\ {}_3 & {}_3^{(i)} & {}_3^+ \\ \boldsymbol{v}^{d,-} & \boldsymbol{v}^{d(i)} & \boldsymbol{v}^{d,+} \end{pmatrix}$$

Step 2: Compute tness of each string by evaluating ($^{(i)}$).

Step 3: Rank the genetic strings where top rank has the minimum cost function ($^{(i)}$).

Step 4: Mate the top pairs of genetic strings to obtain 2 children, such that:

$$^{(ci)} = \begin{pmatrix} -_1^{pi} \;_1 + -_1^{p(i+1)}(1 \quad _1) \\ -_2^{pi} \;_2 + -_2^{p(i+1)}(1 \quad _2) \\ -_3^{pi} \;_3 + -_3^{p(i+1)}(1 \quad _3) \\ \boldsymbol{v}^{d,pi} \;_4 + \quad \boldsymbol{v}^{d,p(i+1)}(1 \quad _4) \end{pmatrix}$$

where $_j$ 2 rand[0,1].

$$^{(c(i+1))} = \begin{pmatrix} -_1^{p(i+1)} \hat{}_1 + -_1^{pi}(1 \quad \hat{}_1) \\ -_2^{p(i+1)} \hat{}_2 + -_2^{pi}(1 \quad \hat{}_2) \\ -_3^{p(i+1)} \hat{}_3 + -_3^{pi}(1 \quad \hat{}_3) \\ \boldsymbol{v}^{d,p(i+1)} \hat{}_4 + \quad \boldsymbol{v}^{d,pi}(1 \quad \hat{}_4) \end{pmatrix}$$

where $\hat{}_j$ 2 rand[0,1].

Step 5: Generate S P P new random genetic strings. Remove bottom S P original strings from population.

Step 6: Repeat steps 2-5 with new population until:

- G generations has been reached.
- $min(\;)$ TOL.

In this model the number of strings used is S = 100, while the number of parents and children are equal to P = 10. The GA is run until an error tolerance of TOL = 0:02 or G = 100 generations is reached. The rest of the parameters used in this optimization process are given in the variable glossary.

Deliverables

PART 1: FULL MODEL WITH ELECTRICAL FORCES

In the rst part of this assignment we model the droplets' dynamics, with droplets deposited for each of N_t time steps, not including t = 0, for a total droplet count $N_d = N_t$: We place several point charges on a print surface at y = 0, with a 10-by-10 grid of points each with charge q_p. Center the grid at the origin within a square with sides of length L_{bed}. To generate the grid positions (same for both x and z components), you can use linspace (L_{bed}=2; L_{bed}=2; 10) and meshgrid(). We model the printer arm extruding for a total simulation time T = 3 seconds, such that the time step size $t = T=N_t$ with t = 0:001 seconds. After all the droplets are extruded, continue the simulation with the same time step size until all droplets land on the substrate.

1. For all requested plots below, plot the point charges on the same gure. For visualization, let the boundary created by the point charges de ne the boundary created by the print bed.
2. Provide a plot of the nal droplet pattern in a "top-down" (z x) view. Color each droplet according its ight time (i.e. time in the air) using scatter and show the color axis with colorbar. Be sure to label your color-bar with title and units.
3. Provide a 3D plot of the tool path of the robot arm's dispenser end with the beginning and end of the tool-path marked on the same plot as the print pattern on the print bed with the rst and last droplet marked. Be sure to label these points using a legend.
4. Explain why you think the resulting droplet pattern occurred.
5. Set the electrical force to 0. Compare the animations and nal con gurations of the droplets with and without the electrical force. What role does the electrically charged substrate have on the motion of the droplets? It is up to you how to present this comparison, but be sure it is clear in your report. What adversary force does the electrical force assist against?

PART 2: SIMPLIFIED FREE FALL MODEL WITH GENETIC ALGORITHM

For the simpli ed model, we change the following:

- Remove drag from the model.
- Remove the electrical forces, leaving only gravity.

In doing so, our droplet dynamics reduces to a free fall model. The extruder model does not change from Part 1 . Because our droplets only move by gravity, they are under constant acceleration, and thus the nal position of the droplets can be solved for analytically. You should solve for that analytical solution and use it in your code. Do not use time stepping for this part of the project.

Here, we provide a data le robotprintdata.mat which contains a variable *r* i, a 3 N_d array where each column is a droplet's nal position components with the columns ordered according to extrusion time. The aim of your genetic algorithm is to try and reproduce the design string ^{des} that produced the desired droplet positions r_i^{des} by minimizing . It is your discretion how low to set your cost tolerance based on physical performance. You will not be evaluated on the nal value of your cost function but rather if it converges and the pattern follows relatively closely to the desired pattern.

1. Analytically solve for the nal position of the droplets on the print if they are in free fall. Show the steps to your derivation. Hint: Use constant acceleration kinematic equations from physics.
2. Describe the parameters you used in your GA. Use S = 100 genetic strings, P = 10 parents which generate P = 10 children in the same ordering as previous projects.
3. Provide a z x "top-down" plot of the nal droplet pattern on the same plot as the desired droplet pattern provided in the data le. How well does your design string follow the desired pattern?
4. Provide a convergence plot showing the total cost of the best design, the mean of all parent designs, and the mean of the overall population for each generation.
5. Report your best-performing 4 designs in a table.

VARIABLE GLOSSARY

Table 3.1: Model Parameters.

Symbol	Units	Value	Description
r_0	m	(0, 0.5, 0)	Fixed arm end position
$\Theta_1^0, \Theta_2^0, \Theta_3^0$	rad	$\pi/2$, 0, 0	Initial rod angles
$\dot{\Theta}_1^0, \dot{\Theta}_2^0, \dot{\Theta}_3^0$	rad/s	0.2,-0.2,10	Link 1,2,3 angular velocity
L_{bed}	m	0.8	Charged grid side length
L_1, L_2, L_3	m	0.3, 0.2, 0.08	Rod lengths
g	m/s^2	9.81	Gravitational acceleration
ϵ	F/m	8.854 10^{12}	Electric permittivity
R	m	0.001	Droplet radius
v_2	-	0.25	Phase 2 volume fraction
ρ_1	kg/m^3	2000	Phase 1 density
rho_2	kg/m^3	7000	Phase 2 density
q_1	C/m^3	0	Phase 1 charge capacity
q_2	C/m^3	1e-3	Phase 2 charge capacity
Δv^d	m/s	(0, -1.2, 0)	Relative extrusion velocity
q_p	C	8 105	Grid pixel charge
v^f	m/s	(0.5, 0, 0.5)	Surrounding medium velocity
ρ_a	kg/m^3	1.225	Surrounding medium density
μ_f	Pa/s	18 10	Surrounding medium viscosity
Δt	s	0.001	Time Step Size
T	s	3	Total Simulation Time
$\dot{\Theta}_1, \dot{\Theta}_1^+$	rad/s	[15, 16]	Search bounds for $\dot{\Theta}_1$
$\dot{\Theta}_2, \dot{\Theta}_2^+$	rad/s	[15, 16]	Search bounds for $\dot{\Theta}_2$
$\dot{\Theta}_3, \dot{\Theta}_3^+$	rad/s	[6, 7]	Search bounds for $\dot{\Theta}_3$
$\Delta v^{d,}$, $\Delta v^{d,+}$	m/s	[-3.5, -3]	Search bounds for Δv^d

4 Solution

The assignment solution is encoded in Matlab below.

```
%% Part 1 - Full Model with Electrical Forces

close all;clear all;clc; %house-keeping

% Video start
tic
% v = VideoWriter('robotic.avi');
% open(v);

% Variables

%time
T = 3; %total simulation time [s] - exclude t=0
del_t = 0.001; %time step size [s]
N_t = T / del_t; %number of time steps
%values for bed + arm
L_bed = 0.8; %charged grid side length [m]
r_0 = [0,0.5,0]; %Fixed arm end position [m]
theta_0 = [pi/2,0,0]; %Initial rod angles [rad]
thetad = [0.2,-0.2,10]; %Link 1,2,3 angular velocity [rad/s]
L_rod = [0.3,0.2,0.08]; %Rod lengths [m]

%droplet
N_d = N_t; %number of droplets
R = 0.001; %droplet radius [m]

V_i = (4/3*pi*R^3); %droplet volume [m3]
v_2 = 0.25; %phase 2 volume fraction
rho_1 = 2000; %phase 1 density [kg/m3]
rho_2 = 7000; %phase 2 density [kg/m3]
q_1 = 0; %phase 1 charge capacity [C/m3]
q_2 = 1e-3; %phase 2 charge capacity [C/m3]
del_v = [0, -1.2, 0]; %relative extrusion velocity [m/s]
%effective properties
rho_s = (1-v_2)*rho_1 + v_2*rho_2; %effective density [kg/m3]
q_s = (1-v_2)*q_1 + v_2*q_2; %effective charge capacity[C/m3]
%droplet charge
q_i = V_i * q_s; %droplet charge [C]
%droplet mass
m_i = V_i * rho_s; %droplet mass [kg]

N_t = round(1.2*N_t); %add time for particle settling


%gravity
g = -9.81; %gravitational acceleration [m/s2]
F_g = [0, g*m_i, 0]; %gravity force [N]


%e+m grid
```

```
x_grid = linspace(-L_bed/2,L_bed/2,10);
z_grid = x_grid;
[X_grid, Z_grid] = meshgrid(x_grid, z_grid); %create grid of point charges
Y_grid = zeros(10,10);
r_p = [X_grid(:), Y_grid(:), Z_grid(:)]; %point charge positions [m]

%e-m charge
eps = 8.854e-12; %electric permittivity [F/m]
q_p = -8e-5; %grid pixel charge [C]

%drag
v_f = [0.5, 0, 0.5]; %surrounding medium velocity [m/s]
rho_a = 1.225; %surrounding medium density [kg/m3]
mu_f = 1.8e-5; %surrounding medium viscosity [Pa/s]
A_i = pi*R^2; %drag reference area[m2]

% Robotic Arm Dynamics

theta = theta_0;

%pre-allocations
x_d = [];
y_d = [];
z_d = [];
r_drop = NaN(N_d,3);
v_drop = NaN(N_d,3);
r_disp = NaN(1,3);
v_disp = NaN(1,3);
F_tot = NaN(1,3);
F_elec = NaN(1,3);
F_drag = NaN(1,3);

%color plot code
c = linspace(1,10,N_d);
t_fly = NaN(N_d,1);


for i= 1:N_t
    if i<=N_d
        %%%%% Robotic Arm Dynamics %%%%%%%%%%%%%%%%%%%
        %dispenser position
        theta = theta + thetad*del_t;
        x_d = L_rod(1)*cos(theta(1)) + L_rod(2)*cos(theta(2)) + L_rod(3)*sin(
            theta(3));
        y_d = L_rod(1)*sin(theta(1)) + L_rod(2)*sin(theta(2));
        z_d = L_rod(3)*cos(theta(3));
        r_disp = r_0 + [x_d,y_d,z_d]; %dispenser position

        r_drop(i,:) = r_disp;

        %robot arm location
        x_a1 = r_0(1) + L_rod(1)*cos(theta(1));
        y_a1 = r_0(2) + L_rod(1)*sin(theta(1));
        x_a2 = x_a1 + L_rod(2)*cos(theta(2));
```

```
105         y_a2 = y_a1 + L_rod(2)*sin(theta(2));
106         x_a = [0,x_a1,x_a2,r_disp(1)]; % x-coord of robot arm
107         y_a = [0,y_a1,y_a2,r_disp(2)]; % y-coord of robot arm
108         z_a = [0,0,0,r_disp(3)]; % z-coord of robot arm
109
110         %dispenser velocity
111         xd_d = -L_rod(1)*thetad(1)*sin(theta(1)) + -L_rod(2)*thetad(2)*sin(
                theta(2)) + L_rod(3)*thetad(3)*cos(theta(3));
112         yd_d = L_rod(1)*thetad(1)*cos(theta(1)) + L_rod(2)*thetad(2)*cos(theta
                (2));
113         zd_d = -L_rod(3)*thetad(3)*sin(theta(3));
114         v_disp = [xd_d,yd_d,zd_d]; %dispenser velocity
115
116         v_drop(i,:) = v_disp + del_v;
117     end
118
119     %%%% Droplet Dynamics %%%%%%%%%%%%%%%%%%
120
121
122
123     %forward euler
124     if i<=N_d
125         for k = 1:i
126             if r_drop(k,2)<=0
127                 r_drop(k,2)=0;
128                 if isnan(t_fly(k))
129                     t_fly(k)=i;
130                 end
131             else
132                 %e+m
133                 F_elec = [0 0 0]; %reset elec force
134                 for m = 1:numel(r_p(:,1))
135                     F_elec = F_elec + (q_p * q_i)/(4*pi*eps*(norm(r_drop(k,:)-
                            r_p(m,:)))^3)*(r_drop(k,:)-r_p(m,:));
136                 end
137                 %drag
138                 Re = (2*R*rho_a*norm(v_f-v_drop(k,:)))/(mu_f);
139                 if Re<1
140                     CD_i = Re/24;
141                 elseif Re<=400
142                     CD_i = 24/Re^0.0646;
143                 elseif Re<=3e5
144                     CD_i = 0.5;
145                 elseif Re<=2e6
146                     CD_i = 0.000366*Re^0.4275;
147                 else
148                     CD_i = 0.18;
149                 end
150                 F_drag = (1/2)*rho_a*CD_i*norm(v_f-v_drop(k,:))*(v_f-v_drop(k
                        ,:))*A_i;
151                 %total force
152                 F_tot = F_g + F_drag + F_elec;
153                 r_drop(k,:) = r_drop(k,:) + del_t .* v_drop(k,:);
154                 v_drop(k,:) = v_drop(k,:) + del_t./m_i .* F_tot;
```

```
155             end
156         end
157     else
158         for k = 1:N_d
159             if r_drop(k,2)<=0
160                 r_drop(k,2)=0;
161                 if isnan(t_fly(k))
162                     t_fly(k)=i;
163                 end
164             else
165                 %e+m
166                 F_elec = [0 0 0]; %reset elec force
167                 for m = 1:numel(r_p(:,1))
168                     F_elec = F_elec + (q_p * q_i)/(4*pi*eps*(norm(r_drop(k,:)-
                        r_p(m,:)))^3)*(r_drop(k,:)-r_p(m,:));
169                 end
170                 %drag
171                 Re = (2*R*rho_a*norm(v_f-v_drop(k,:)))/(mu_f);
172                 if Re<1
173                     CD_i = Re/24;
174                 elseif Re<=400
175                     CD_i = 24/Re^0.0646;
176                 elseif Re<=3e5
177                     CD_i = 0.5;
178                 elseif Re<=2e6
179                     CD_i = 0.000366*Re^0.4275;
180                 else
181                     CD_i = 0.18;
182                 end
183                 F_drag = (1/2)*rho_a*CD_i*norm(v_f-v_drop(k,:))*(v_f-v_drop(k
                    ,:))*A_i;
184                 %total force
185                 F_tot = F_g + F_drag + F_elec;
186                 r_drop(k,:) = r_drop(k,:) + del_t .* v_drop(k,:);
187                 v_drop(k,:) = v_drop(k,:) + del_t./m_i .* F_tot;
188             end
189         end
190     end
191
192
193
194     %%%% Plotting %%%%%%%%%%%%%%%%%%%%%%%%
195
196     if mod(i,10) == 0 % update movie frame
197     figure(1)
198
199     scatter3(r_drop(:,1), r_drop(:,3), r_drop(:,2),[],c)
200     %scatter(x,y,[],c)
201     hold on
202     plot3(x_a, z_a, y_a,'o-','Color','r','LineWidth',3)
203     scatter3(r_p(:,1), r_p(:,3), r_p(:,2),'d','m','filled')
204     xlim([-L_bed/2, L_bed/2]);
205     ylim([-L_bed/2, L_bed/2]);
206     zlim([0, 2]);
```

```
207         xlabel('X[m]');
208         ylabel('Z[m]');
209         zlabel('Y[m]');
210         view(45, 45);
211         grid on;
212         title('Full Model')
213         cb = colorbar ; %Create Colorbar
214         cb.Ticks = linspace(1, 10, 2) ;
215         cb.TickLabels =  {'old','new'} ;
216         set(get(cb,'title'),'string','Particle Deposition');
217 %           frame = getframe(gcf);
218 %           writeVideo(v,frame);
219         end
220
221
222 %       if i == N_t % figures for t=1,2,3,final
223 %       figure(1)
224 %       scatter3(r_drop(:,1), r_drop(:,3), r_drop(:,2),[],c)
225 %       %scatter(x,y,[],c)
226 %       hold on
227 %       plot3(x_a, z_a, y_a,'o-','Color','r','LineWidth',3)
228 %       scatter3(r_p(:,1), r_p(:,3), r_p(:,2),'d','m','filled')
229 %       xlim([-L_bed/2, L_bed/2]);
230 %       ylim([-L_bed/2, L_bed/2]);
231 %       zlim([0, 2]);
232 %       xlabel('X[m]');
233 %       ylabel('Z[m]');
234 %       zlabel('Y[m]');
235 %       view(45, 45);
236 %       grid on;
237 %       title('Full Model')
238 %       cb = colorbar ; %Create Colorbar
239 %       cb.Ticks = linspace(1, 10, 2) ;
240 %       cb.TickLabels =  {'old','new'} ;
241 %       set(get(cb,'title'),'string','Particle Deposition');
242 %       end
243
244         hold off
245 end
246
247 % t_fly = t_fly - [1:1:N_d]'; % figure for flight time
248 %
249 % t_fly_act = t_fly □ del_t;
250 % %
251 % c = linspace(min(t_fly_act),max(t_fly_act),length(t_fly_act));
252 % figure(1) % figure for droplet pattern w/ flight time
253 % scatter3(r_drop(:,1), r_drop(:,3), t_fly_act,[],c)
254 % hold on
255 % %plot3(x_a, z_a, y_a,'o-','Color','r','LineWidth',3)
256 % scatter3(r_p(:,1), r_p(:,3), r_p(:,2),'d','m','filled')
257 % xlim([-L_bed/2, L_bed/2]);
258 % ylim([-L_bed/2, L_bed/2]);
259 % zlim([0, 2]);
260 % xlabel('X[m]');
```

```
% ylabel('Z[m]');
% zlabel('Y[m]');
% view(90, 270);
% grid on;
% title('Full Model')
% cb = colorbar ; %Create Colorbar
% %cb.Ticks = linspace(1, 10, 2) ;
% %cb.TickLabels =  {'old','new'} ;
% set(get(cb,'title'),'string','Flight Time [s]');



% frame = getframe(gcf); %video write
% writeVideo(v,frame);
% hold off
% close(v);
toc;

%% Part 2 - Simplified Free Fall Model with Genetic Algorithm

close all;clear all;clc; %house-keeping

tic
% Variables

%import data file
load('robotprint_data.mat', 'ri')
ri = ri';


%time
T = 3; %total simulation time [s] - exclude t=0
del_t = 0.001; %time step size [s]
N_t = T / del_t; %number of time steps
%values for bed + arm
L_bed = 0.8; %charged grid side length [m]
r_0 = [0,0.5,0]; %Fixed arm end position [m]
theta_0 = [pi/2,0,0]; %Initial rod angles [rad]
%thetad = [0.2,-0.2,10]; %Link 1,2,3 angular velocity [rad/s] - OPTIMIZED
L_rod = [0.3,0.2,0.08]; %Rod lengths [m]

%droplet
N_d = N_t; %number of droplets
R = 0.001; %droplet radius [m]

V_i = (4/3*pi*R^3); %droplet volume [m3]
v_2 = 0.25; %phase 2 volume fraction
rho_1 = 2000; %phase 1 density [kg/m3]
rho_2 = 7000; %phase 2 density [kg/m3]
%del_v = [0, -1.2, 0]; %relative extrusion velocity [m/s] - OPTIMIZED
%effective properties
rho_s = (1-v_2)*rho_1 + v_2*rho_2; %effective density [kg/m3]
%droplet mass
m_i = V_i * rho_s; %droplet mass [kg]
```

```
315
316
317 %gravity
318 g = -9.81; %gravitational acceleration [m/s2]
319 F_g = [0, g.*m_i, 0]; %gravity force [N]
320
321 %total force
322 F_tot = F_g;
323
324 %%%%%GENETIC ALGORITHM START%%%%%%%%
325
326 %general GA parameters
327 S = 100; % # of genetic strings
328 P = 10; % # of parents
329 C = 10; % # of children
330 TOL = 0.02; % error tolerance
331
332 %%Step 1: Create S design strings
333 thetad = zeros(S,3);
334 del_v = zeros(S,3);
335
336 thetad(:,1) = 15 + (16-15).*rand(S,1);
337 thetad(:,2) = 15 + (16-15).*rand(S,1);
338 thetad(:,3) = 6 + (7-6).*rand(S,1);
339 del_v(:,2) = -3.5 + (-3-(-3.5)).*rand(S,1);
340
341 time = [del_t:del_t:T];
342
343 %pre-allocation
344 r_drop_all = NaN(N_d,3,S);
345 Pi = NaN(S,1);
346
347 n_gen = 0; %number of generations counter
348 j = 1; %while loop counter
349 Pi(1)= 1000; %high Pi to start while loop
350
351 while min(Pi)>TOL
352     for k = 1:S %string loop
353     %determine dispenser position & velocity for each droplet
354     theta = theta_0 + thetad(k,:).*time(:);
355     x_d = L_rod(1).*cos(theta(:,1)) + L_rod(2).*cos(theta(:,2)) + L_rod(3).*
          sin(theta(:,3));
356     y_d = L_rod(1).*sin(theta(:,1)) + L_rod(2).*sin(theta(:,2));
357     z_d = L_rod(3).*cos(theta(:,3));
358     r_disp = r_0 + [x_d,y_d,z_d]; %dispenser position
359
360     %dispenser velocity
361     xd_d = -L_rod(1).*thetad(k,1).*sin(theta(:,1)) + -L_rod(2).*thetad(k,2).*sin(
          theta(:,2)) + L_rod(3).*thetad(k,3).*cos(theta(:,3));
362     yd_d = L_rod(1).*thetad(k,1).*cos(theta(:,1)) + L_rod(2).*thetad(k,2).*cos(
          theta(:,2));
363     zd_d = -L_rod(3).*thetad(k,3).*sin(theta(:,3));
364     v_disp = [xd_d,yd_d,zd_d]; %dispenser velocity
365
```

```
366     %analytical solution for particle final position
367     t_fly = (-(v_disp(:,2)+del_v(k,2))-sqrt((v_disp(:,2)+del_v(k,2)).^2-2*
            F_tot(2)./m_i.*r_disp(:,2))).*(F_tot(2)./m_i);
368     r_drop = F_tot./(2*m_i).*t_fly.^2 + (v_disp+del_v(k,:)).*t_fly + r_disp;
369     r_drop(:,2) = 0; %correction for truncation error when t_fly plugged in
370
371     %record droplet position of all strings
372     r_drop_all(:,:,k) = r_drop;
373
374     errNum = 0; % initialize error numerator as zero
375     errDen = 0; % initialize error denumerator as zero
376     %%Step 2: Evaluate fitness of strings
377     r_diff = ri-r_drop;
378
379     for m = 1:N_d
380         errNum_e = norm(r_diff(m,:));
381         errDen_e = norm(ri(m,:));
382         errNum = errNum + errNum_e;
383         errDen = errDen + errDen_e;
384     end
385     Pi(k) = errNum/errDen;
386     end
387
388     %%Step 3: Rank the strings by ascending value of Pi
389     [Pi,ind] = sort(Pi); % keep the original indices for finding associated
            Lam,uh
390     r_drop_all = r_drop_all(:,:,ind); %align des strings and sol vectors in
            order it was sorted above
391     thetad = thetad(ind,:);
392     del_v = del_v(ind,:);
393
394     %%Step 4: Mate top strings
395     %mate for thetad
396     phi = rand(P,3); % calculate phi
397     ind1 = [1:2:P]'; % child 1 index pair
398     ind2 = [2:2:P]'; % child 2 index pair
399     thetad = vertcat(thetad(1:P,:),phi(ind1).*thetad(ind1,:)+(1-phi(ind1)).*
            thetad(ind2,:),...
400          phi(ind2).*thetad(ind2,:)+(1-phi(ind2)).*thetad(ind1,:)); %
                concatenate c on p
401     %mate for del_v
402     phi = rand(P,1); % calculate phi
403     del_v = vertcat(del_v(1:P,:),phi(ind1).*del_v(ind1,:)+(1-phi(ind1)).*del_v
            (ind2,:),...
404          phi(ind2).*del_v(ind2,:)+(1-phi(ind2)).*del_v(ind1,:)); % concatenate
                c on p
405
406     %%Step 5: Generate S-2P new strings
407
408     thetad = vertcat(thetad,[15 + (16-15).*rand(S-2*P,1), 15 + (16-15).*rand(S
            -2*P,1), 6 + (7-6).*rand(S-2*P,1)]);
409     del_v = vertcat(del_v,[zeros(S-2*P,1), -3.5 + (-3-(-3.5)).*rand(S-2*P,1),
            zeros(S-2*P,1)]);
410
```

```
411      %Save generation-based outputs
412      Min(j)=min(Pi);
413      Ave(j)=mean(Pi);
414      Ave_parent(j)=mean(Pi(1:P));
415
416      n_gen = n_gen + 1; %number of generations counter
417      j = j + 1; %while loop counter
418
419      if n_gen >= 100
420          break
421      end
422  end
423
424
425
426
427  %%%% Gen Plotting %%%%%%%%%%%%%%%%%%%%%%%%
428  figure(1)
429  plot(1:n_gen,Min,'b',1:n_gen,Ave,'r',1:n_gen,Ave_parent,'g')
430  legend('Best Design','Mean of Population','Mean of Parents')
431  xlabel('Generation')
432  ylabel('Error')
433  title('GA Results')
434  xtickformat('%.0f')
435  hold off
436
437  %%%%Obtain Best Design Parameters%%%%%%%%%
438  Lambda = horzcat(thetad(1,:), del_v(1,2));
439
440  r_drop_best = r_drop_all(:,:,1);
441  %%%% Plotting %%%%%%%%%%%%%%%%%%%%%%%%
442      figure(2)
443      plot3(r_drop_best(:,1), r_drop_best(:,3), r_drop_best(:,2),'--','Color','r
         ')
444      %scatter3(r_drop_best(:,1), r_drop_best(:,3), r_drop_best(:,2),'b','filled
         ')
445      hold on
446      plot3(ri(:,1), ri(:,3), ri(:,2),'-','Color','b')
447      %scatter3(r_drop_best(:,1), r_drop_best(:,3), r_drop_best(:,2),'b','filled
         ')
448      %xlim([-L_bed/2, L_bed/2]);
449      %ylim([-L_bed/2, L_bed/2]);
450      zlim([0, 2]);
451      xlabel('X[m]');
452      ylabel('Z[m]');
453      zlabel('Y[m]');
454      view(90, 270);
455      grid on;
456      hold off
457  toc;
458
459  figure(3)
460  comparepattern(Lambda(1),Lambda(2),Lambda(3),Lambda(4))
461
```

```
%% EXTRA - Simulate best design
% Video start
tic
% v = VideoWriter('roboticGA.avi');
% open(v);

% Variables

%time
T = 3; %total simulation time [s] - exclude t=0
del_t = 0.001; %time step size [s]
N_t = T / del_t; %number of time steps
%values for bed + arm
L_bed = 0.8; %charged grid side length [m]
r_0 = [0,0.5,0]; %Fixed arm end position [m]
theta_0 = [pi/2,0,0]; %Initial rod angles [rad]
thetad = Lambda(1:3); %Link 1,2,3 angular velocity [rad/s]
L_rod = [0.3,0.2,0.08]; %Rod lengths [m]

%droplet
N_d = N_t; %number of droplets
R = 0.001; %droplet radius [m]

V_i = (4/3*pi*R^3); %droplet volume [m3]
v_2 = 0.25; %phase 2 volume fraction
rho_1 = 2000; %phase 1 density [kg/m3]
rho_2 = 7000; %phase 2 density [kg/m3]
q_1 = 0; %phase 1 charge capacity [C/m3]
q_2 = 1e-3; %phase 2 charge capacity [C/m3]
del_v = [0, Lambda(4), 0]; %relative extrusion velocity [m/s]
%effective properties
rho_s = (1-v_2)*rho_1 + v_2*rho_2; %effective density [kg/m3]
q_s = (1-v_2)*q_1 + v_2*q_2; %effective charge capacity[C/m3]
%droplet charge
q_i = V_i * q_s; %droplet charge [C]
%droplet mass
m_i = V_i * rho_s; %droplet mass [kg]

N_t = round(1.2*N_t); %add time for particle settling


%gravity
g = -9.81; %gravitational acceleration [m/s2]
F_g = [0, g*m_i, 0]; %gravity force [N]

% Robotic Arm Dynamics

theta = theta_0;

%pre-allocations
x_d = [];
y_d = [];
z_d = [];
r_drop = NaN(N_d,3);
```

```
v_drop = NaN(N_d,3);
r_disp = NaN(1,3);
v_disp = NaN(1,3);
F_tot = NaN(1,3);


for i= 1:N_t
    if i<=N_d
        %%%%% Robotic Arm Dynamics %%%%%%%%%%%%%%%%%%%%
        %dispenser position
        theta = theta + thetad*del_t;
        x_d = L_rod(1)*cos(theta(1)) + L_rod(2)*cos(theta(2)) + L_rod(3)*sin(
            theta(3));
        y_d = L_rod(1)*sin(theta(1)) + L_rod(2)*sin(theta(2));
        z_d = L_rod(3)*cos(theta(3));
        r_disp = r_0 + [x_d,y_d,z_d]; %dispenser position

        r_drop(i,:) = r_disp;

        %robot arm location
        x_a1 = r_0(1) + L_rod(1)*cos(theta(1));
        y_a1 = r_0(2) + L_rod(1)*sin(theta(1));
        x_a2 = x_a1 + L_rod(2)*cos(theta(2));
        y_a2 = y_a1 + L_rod(2)*sin(theta(2));
        x_a = [0,x_a1,x_a2,r_disp(1)]; % x=coord of robot arm
        y_a = [0,y_a1,y_a2,r_disp(2)]; % y=coord of robot arm
        z_a = [0,0,0,r_disp(3)]; % z=coord of robot arm

        %dispenser velocity
        xd_d = =L_rod(1)*thetad(1)*sin(theta(1)) + =L_rod(2)*thetad(2)*sin(
            theta(2)) + L_rod(3)*thetad(3)*cos(theta(3));
        yd_d = L_rod(1)*thetad(1)*cos(theta(1)) + L_rod(2)*thetad(2)*cos(theta
            (2));
        zd_d = =L_rod(3)*thetad(3)*sin(theta(3));
        v_disp = [xd_d,yd_d,zd_d]; %dispenser velocity

        v_drop(i,:) = v_disp + del_v;
    end

    %%%%% Droplet Dynamics %%%%%%%%%%%%%%%%%%%%


    %forward euler
    if i<=N_d
        for k = 1:i
            if r_drop(k,2)<=0
                r_drop(k,2)=0;
            else
                F_tot = F_g;
                r_drop(k,:) = r_drop(k,:) + del_t .* v_drop(k,:);
                v_drop(k,:) = v_drop(k,:) + del_t./m_i .* F_tot;
            end
        end
```

```
567         else
568             for k = 1:N_d
569                 if r_drop(k,2)<=0
570                     r_drop(k,2)=0;
571                 else
572                     %total force
573                     F_tot = F_g;
574                     r_drop(k,:) = r_drop(k,:) + del_t .* v_drop(k,:);
575                     v_drop(k,:) = v_drop(k,:) + del_t./m_i .* F_tot;
576                 end
577             end
578         end
579         %%%% Plotting %%%%%%%%%%%%%%%%%%%%%%%%%
580
581         if mod(i,10) == 0 % update movie frame
582         figure(4)
583
584         scatter3(r_drop(:,1), r_drop(:,3), r_drop(:,2),'r','filled')
585         hold on
586         plot3(x_a, z_a, y_a,'o-','Color','g','LineWidth',3)
587         plot3(ri(:,1), ri(:,3), ri(:,2),'-','Color','b')
588         xlim([-2, 2.5]);
589         ylim([-L_bed/2, L_bed/2]);
590         zlim([min(ri(:,3)), 2+min(ri(:,3))]);
591         xlabel('X[m]');
592         ylabel('Z[m]');
593         zlabel('Y[m]');
594         view(225, 45);
595         %view(90, 270);
596         grid on;
597         title('Full Model')
598 %           frame = getframe(gcf);
599 %           writeVideo(v,frame);
600         end
601         hold off
602
603 %           if i == N_t % figures for t=1,2,3,final
604 %               figure(1)
605 %               scatter3(r_drop(:,1), r_drop(:,3), r_drop(:,2),'r','filled')
606 %               hold on
607 %               plot3(x_a, z_a, y_a,'o-','Color','g','LineWidth',3)
608 %               plot3(ri(:,1), ri(:,3), ri(:,2),'-','Color','b')
609 %               xlim([-2, 2.5]);
610 %               ylim([-L_bed/2, L_bed/2]);
611 %               zlim([min(ri(:,3)), 2+min(ri(:,3))]);
612 %               xlabel('X[m]');
613 %               ylabel('Z[m]');
614 %               zlabel('Y[m]');
615 %               view(225, 45);
616 %               grid on;
617 %               title('Full Model')
618 %           end
619
620 end
```

```
621
622
623 % frame = getframe(gcf);
624 % writeVideo(v,frame);
625 % hold off
626 % close(v);
627 toc;
```

5 Ethical Considerations for this Project

A goal of this project is to enable advancements in science and engineering through to address critical national challenges associated with next generation food systems. There are deep ethical considerations associated with any technology, in particular for food systems. While technology has tremendous potential to identify greater efficiencies, when it is created without appropriate consideration for who will have access to and control over new resources, or how the new technologies will impact those who work in the system, the efficiencies identified may come at the cost of greater societal inequity. It is important to pursue harnessing technology to disrupt existing inequities, rather than further entrench existing power structures. The following areas should be considered:

- Labor: 1) occupational health, 2) food manufacturing, and 3) outdoor agriculture labor;
- Producers: 1) Small- to mid-size farms, 2) urban agriculture, and 3) research in farm transitions; Technology: 1) research in technology and democracy;
- Health Human Rights: 1) land rights, 2) social justice, and 3) decolonization in agriculture;

Please consider the following questions:

- What are the societal implications of the technology that you are developing?
- Can this technology be distributed fairly and equitably to a wide variety of entities in agricultural industry?
- Are there any potential unintended consequences of this technology becoming available?
- Are there any harmful "spinoffs" of this technology?
- Are there any useful "spinoffs" of this technology?

6 References

1. Zohdi, T. I. and Dornfeld D. A. (2015) Future Synergy between Computational Mechanics and Advanced Additive Manufacturing. US National Academies Report:
   http : ==sites:nationalacademies:org=cs=groups=pgasite=documents=webpage=pga$_1$*66813:pdf*
2. Huang, Y. Leu, M. C., Mazumdar, J. and Donmez, A. (2015). Additive manufacturing: current state, future potential, gaps and needs, and recommendation. *Journal of Manufacturing Science and Engineering.* vol 137, 014001-1.
3. Hunt, K. H. (1979). Kinematic Geometry of Mechanisms, Oxford Engineering Science Series.
4. Hartenberg, R. S. and Denavit, J. (1964). Kinematic Synthesis of Linkages, McGraw-Hill, New York.
5. Howell, L. L. (2001). Compliant mechanisms, John Wiley & Sons.
6. McCarthy, J. M. and Soh, G. S. (2010). Geometric Design of Linkages, Springer, New York.
7. McCarthy, J. M. (1990). Introduction to Theoretical Kinematics, MIT Press, Cambridge, MA.
8. Reuleaux, F., (1876). The Kinematics of Machinery, (trans. and annotated by A. B. W. Kennedy), reprinted by Dover, New York (1963).
9. Sandor, G.N. and Erdman,A.G. (1984). Advanced Mechanism Design: Analysis and Synthesis, Vol. 2. Prentice-Hall, Englewood Cli s, NJ.
10. Slocum, A. (1992). Precision Machine Design, SME
11. Suh, C. H. and Radcli e, C. W. (1978). Kinematics and Mechanism Design, John Wiley and Sons, New York.
12. Uicker, J. J., Pennock, G. R. and Shigley, J. E. (2003). Theory of Machines and Mechanisms, Oxford University Press, New York.
13. Papageorgiou, D. T. (1995). On the breakup of viscous liquid threads. Physics of Fluids. 7 (7): 1529-1521.
14. Eggers, J. (1997). Nonlinear dynamics and breakup of free-surface ows. Reviews of Modern Physics. 69 (3): 865.
15. Zohdi, T. I. (2017). Laser-induced heating of dynamic depositions in additive manufacturing. Comp. Meth. in Appl. Mech. and Eng. https://doi.org/10.1016/j.cma.2017.11.003
16. Zohdi, T. I. (2017).On simple scaling laws for pumping uids with electrically-charged particles. Int. Journal of Eng. Sci: https://doi.org/10.1016/j.ijengsci.2017.11.003
17. Jackson, J. D. (1998). Classical Electrodynamics. Third Edition. Wiley.
18. Chow, C. Y. (1980). An introduction to computational uid dynamics. New York, Wiley.
19. Schlichtling, H. (1979). Boundary-layer theory. 7th edition. New York: McGraw-Hill.
20. Whitaker, S. (1972) Forced convection heat transfer correlations for ow in pipes, past at plates, single cylinders, single spheres, and ow in packed beds and tube bundles. AIChE J. 18, 361{371.
21. Hashin, Z. and Shtrikman, S. (1962) On some variational principles in anisotropic and nonhomogeneous elasticity. Journal of the Mechanics and Physics of Solids. **10**, 335-342.
22. Hashin, Z. and Shtrikman, S. (1963) A variational approach to the theory of the elastic behaviour of multiphase materials. Journal of the Mechanics and Physics of Solids. **11**, 127-140.
23. Hashin, Z. (1983) Analysis of composite materials: a survey. ASME Journal of Applied Mechanics. **50**, 481-505.
24. Torquato, S. (2002) Random Heterogeneous Materials: Microstructure and Macroscopic Properties Springer-Verlag, New York.
25. Jikov, V. V., Kozlov, S. M., Olenik, O. A.(1994) Homogenization of di erential operators and integral functionals. Springer-Verlag.
26. Mura, T. (1993) Micromechanics of defects in solids, 2nd edition. Kluwer Academic Publishers.
27. Markov, K. Z. 2000. Elementary micromechanics of heterogeneous media. In Heterogeneous Media: Micromechanics Modeling Methods and Simulations (K. Z. Markov, and L. Preziozi, Eds.), pp. 1 162. Birkhauser, Boston.
28. Ghosh, S. (2011). Micromechanical Analysis and Multi-Scale Modeling Using the Voronoi Cell Finite Element Method. CRC Press/Taylor & Francis.
29. Ghosh, S. and Dimiduk, D. (2011). Computational Methods for Microstructure-Property Relations. Springer NY.
30. Matous, K. Geers, M., Kouznetsova, V. G. and Gillman, A. (in press). A review of predictive nonlinear theories for multiscale modeling of heterogeneous materials. *Journal of Computational Physics*.
31. Zohdi, T. I. and Wriggers, P. (*Book*, 2005, 2008) Introduction to computational micromechanics. Second Reprinting (*Peer Reviewed*). Springer-Verlag.
32. Zohdi, T. I. (*Book*, 2012). Electromagnetic properties of multiphase dielectrics. A primer on modeling, theory and computation. Springer-Verlag.
33. Zohdi, T. I. (2002). An adaptive-recursive staggering strategy for simulating multi eld coupled processes in microheterogeneous solids. *The International Journal of Numerical Methods in Engineering.* **53**, 1511-1532.

34. Zohdi, T. I. (2004). Modeling and simulation of a class of coupled thermo-chemo-mechanical processes in multiphase solids. *Computer Methods in Applied Mechanics and Engineering.* Vol. 193/6-8 679-699.
35. Zohdi, T. I. (2004). Modeling and direct simulation of near- eld granular ows. *The International Journal of Solids and Structures.* Vol 42/2 pp 539-564.
36. Zohdi, T. I. (2005). Charge-induced clustering in multi eld particulate ow *The International Journal of Numerical Methods in Engineering.* Volume 62, Issue 7, Pages 870-898
37. Zohdi, T. I. (2007). Computation of strongly coupled multi eld interaction in particle- uid systems. *Computer Methods in Applied Mechanics and Engineering.* Volume 196, 3927-3950.
38. Zohdi, T. I. (2010) On the dynamics of charged electromagnetic particulate jets. *Archives of Computational Methods in Engineering*. Volume 17, Number 2, 109-135
39. Zohdi, T. I. (2010) Simulation of coupled microscale multiphysical- elds in particulate-doped dielectrics with staggered adaptive FDTD. *Computer Methods in Applied Mechanics and Engineering.* Volume 199, 79-101.
40. Zohdi, T. I. (2013) Numerical simulation of charged particulate cluster-droplet impact on electri ed surfaces. *Journal of Computational Physics.* 233, 509-526.
41. Zohdi, T. I. (2014) Additive particle deposition and selective laser processing-a computational manufacturing framework. *Computational Mechanics*. Vol 54, 171-191.
42. Zohdi, T. I. (2014) Embedded electromagnetically sensitive particle motion in functionalized uids. *Computational Particle Mechanics*. Vol 1, 27-45.
43. Zohdi, T. I. (2015). Modeling and simulation of cooling-induced residual stresses in heated particulate mixture depositions. *Computational Mechanics*. Volume 56, 613-630.
44. Zohdi, T. I. (2015). Modeling and e cient simulation of the deposition of particulate ows onto compliant substrates. *The International Journal of Engineering Science.* Volume 99, 74-91. doi:10.1016/j.ijengsci.2015.10.012
45. Zohdi, T. I. (2003). Genetic design of solids possessing a random-particulate microstructure. *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*. Vol: 361, No: 1806, 1021-1043.
46. Zohdi, T. I. (2017). Dynamic thermomechanical modeling and simulation of the design of rapid free-form 3D printing processes with evolutionary machine learning. Computer Methods in Applied Mechanics and Engineering. https://doi.org/10.1016/j.cma.2017.11
47. Holland, J. H. (1975). Adaptation in natural and arti cial systems. Ann Arbor, Mich. University of Michigan Press.
48. Goldberg, D. E. (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley.
49. Davis, L (1991). Handbook of Genetic Algorithms. Thompson Computer Press.
50. Onwubiko, C. (2000) Introduction to engineering design optimization. Prentice Hall.
51. Lagaros, N., Papadrakakis, M. and Kokossalakis, G. (2002). Structural optimization using evolutionary algorithms. *Computers & Structures.* 80, 571-589.
52. Papadrakakis, M., Lagaros, N., Thierauf, G. and Cai, J. (1998a). Advanced solution methods in structural optimisation using evolution strategies, *Engineering Computational Journal*. **15** (1), 12-34.
53. Papadrakakis, M., Lagaros, N. and Tsompanakis, Y. (1998b). Structural optimization using evolution strategies and neutral networks, *Computer Methods in Applied Mechanics and Engineering.* **156**, (1), 309-335.
54. Papadrakakis, M., Lagaros, N. and Tsompanakis, Y. (1999a). Optimization of large-scale 3D trusses using Evolution Strategies and Neural Networks. *Int. J. Space Structures.* **14** (3), 211-223.
55. Papadrakakis, M., Tsompanakis, J. and Lagaros, N. (1999b). Structural shape optimisation using evolution strategies. *Eng. Optimization*. **31**, 515-540.
56. Goldberg, D. E. and Deb, K. (2000) Special issue on Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering. 186 (2-4) 121-124.
57. Thomas, Daniel (24 February 2017). Could 3D bioprinted tissues o er future hope for microtia treatment?. International Journal of Surgery. Retrieved 24 February 2017.
58. Nakashima, Yasuharu; Okazak, Ken; Nakayama, Koichiet; Okada, Seiji; Mizu-uchi, Hideki (January 2017). Bone and Joint Diseases in Present and Future. Fukuoka Igaku Zasshi = Hukuoka Acta Medica. 108 (1): 1-7. ISSN 0016-254X. PMID 29226660.
59. Doyle, Ken (15 May 2014). Bioprinting: From patches to parts. Gen. Eng. Biotechnol. News. 34 (10): 1, 34-35. doi:10.1089/gen.34.10.02.
60. US patent 7051654, Boland, Thomas; Wilson, Jr., William Crisp; Xu, Tao, Ink-jet printing of viable cells, issued 2006-05-30
61. Sha ee, Ashkan; Atala, Anthony (2016-03-01). Printing Technologies for Medical Applications. Trends in Molecular Medicine. 22 (3): 254-265. doi:10.1016/j.molmed.2016.01.003.
62. Ozbolat, Ibrahim T. (2015-07-01). Bioprinting scale-up tissue and organ constructs for transplantation. Trends in Biotechnology. 33 (7): 395-400. doi:10.1016/j.tibtech.2015.04.005.
63. Chua, C.K.; Yeong, W.Y. (2015). Bioprinting: Principles and Applications. Singapore: World Scienti c Publishing Co. p. 296. ISBN 9789814612104. Retrieved 17 February 2016.

64. Harmon, K. (2013). A sweet solution for replacing organs (PDF). Scienti c American. 308 (4): 54-55. doi:10.1038/scienti camerican0413-54. Retrieved 17 February 2016.
65. Murphy, Sean; Atala, Anthony (August 5, 2014). 3D bioprinting of tissues and organs. Nature Biotechnology. 32: 773-85. doi:10.1038/nbt.2958. PMID 25093879.
66. Crawford, M. (May 2013). Creating Valve Tissue Using 3-D Bioprinting. ASME.org. American Society of Mechanical Engineers. Retrieved 17 February 2016.
67. Murphy, S.V.; Skardal, A.; Atala, A. (2013). Evaluation of hydrogels for bio-printing applications. Journal of Biomedical Materials Research Part A. 101A (1): 272-284. doi:10.1002/jbm.a.34326. PMID 22941807.
68. Souza, G. R. et al Three-dimensional tissue culture based on magnetic cell levitation. Nat. Nanotechnol. 5, 291-296 (2010)
69. Haisler, W. L. et al. Three-dimensional cell culturing by magnetic levitation. Nat. Protoc. 8, 1940-1949 (2013)
70. Friedrich, J., Seidel, C., Ebner, R. & Kunz-Schughart, L. A. Spheroid-based drug screen: considerations and practical approach. Nat. Protoc. 4, 309-324 (2009)
71. Seiler, A. E. M. & Spielmann, H. The validated embryonic stem cell test to predict embryotoxicity in vitro. Nat. Protoc. 6, 961-978 (2011)
72. Tseng, H. et al.Assembly of a three-dimensional multitype bronchiole coculture model using magnetic levitation. Tissue Eng. Part C. Methods 19, 665-675 (2013)
73. Tseng, H. et al. A three-dimensional co-culture model of the aortic valve using magnetic levitation. Acta Biomater. 10, 173-182 (2014)
74. Daquinag, A. C., Souza, G. R. & Kolonin, M. G, Adipose tissue engineering in three-dimensional levitation tissue culture system based on magnetic nanoparticles. Tissue Eng. Part C. Methods 19, 336-344 (2013)
75. Timm, D. M. et al. A high-throughput three-dimensional cell migration assay for toxicity screening with mobile device-based macroscopic image analysis. Sci. Rep. 3, 3000 (2013)
76. Gwathmey, J. K., Tsaioun, K. & Hajjar, R. J. Cardionomics: a new integrative approach for screening cardiotoxicity of drug candidates. Expert Opin. Drug Metab. Toxicol. 5, 647-660 (2009)
77. Feynman, Richard P.; Leighton, Robert B.; Sands, Matthew (2006). The Feynman Lectures on Physics 2. ISBN 0-8053-9045-6.
78. Cullity, C. D. Graham (2008). Introduction to Magnetic Materials (2 ed.). Wiley-IEEE Press. p. 103. ISBN 0-471-47741-9.
79. Boyer, Timothy H. (1988). The Force on a Magnetic Dipole. American Journal of Physics 56 (8): 688-692. Bibcode:1988AmJPh.56.688B. doi:10.1119/1.15501.
80. Zienkiewicz, O. C. (1984). Coupled problems & their numerical solution, in R. W. Lewis, P. Bettes & E. Hinton eds *Numerical methods in coupled systems* Wiley, Chichester, 35{58
81. Zienkiewicz, O. C., Paul, D. K. and Chan, A. H. C. (1988). Unconditionally stable staggered solution procedure for soil-pore uid interaction problems. *The International Journal of Numerical Methods in Engineering.* **26**, 1039-1055.
82. Lewis, R. W., Schre er, B. A. and Simoni, L (1992) Coupling versus uncoupling in soil consolidation. Int. J. Num. Anal. Metho. Geomech. **15**, 533-548.
83. Lewis, R. W. and Schre er, B. A. (1998) The nite element method in the static and dynamic deformation and consolidation of porous media. 2nd edition. Wiley press.
84. Park, K. C. & Felippa, C. A. (1983). Partitioned analysis of coupled systems. In *Computational methods for transient analysis.* T. Belytschko & T. J. R. Hughes, editors.
85. Farhat, C., Lesoinne, M. and Maman, N. (1995). Mixed Explicit/Implicit Time Integration of Coupled Aeroelastic Problems: Three-Field Formulation, Geometric Conservation and Distributed Solution. *International Journal for Numerical Methods in Fluids,* Vol. 21, pp. 807-835.
86. Farhat, C. and Lesoinne, M. (2000). Two E cient Staggered Procedures for the Serial and Parallel Solution of Three-Dimensional Nonlinear Transient Aeroelastic Problems. *Computer Methods in Applied Mechanics and Engineering*, Vol. 182, pp. 499-516.
87. Farhat, C., van der Zee, G. & Geuzaine, P. (2006). Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity. *Computer Methods in Applied Mechanics & Engineering*. **195**, 1973-2001.
88. Piperno, S. (1997). Explicit/implicit uid/structure staggered procedures with a structural predictor & uid subcycling for 2D inviscid aeroelastic simulations. *Int. J. Num. Meth. Fluids.* **25**, 1207-1226.
89. Piperno, S., Farhat, C. and Larrouturou, B. (1995). Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems - Part I: Model Problem, Theory, and Two-Dimensional Application. *Computer Methods in Applied Mechanics and Engineering*, Vol. 124, Nos. 1-2, pp. 79-112.
90. Piperno, S. and Farhat, C. (2001). Partitioned Procedures for the Transient Solution of Coupled Aeroelastic Problems - Part II: Energy Transfer Analysis and Three-Dimensional Applications. *Computer Methods in Applied Mechanics and Engineering,* Vol. 190, pp. 3147-3170.

91. Michopoulos, G., Farhat, C. and Fish, J. (2005). Survey on Modeling and Simulation of Multiphysics Systems, *Journal of Computing and Information Science in Engineering* Vol 5, Issue 3, 198-213.
92. Lesoinne, M. and Farhat, C. (1998). Free Staggered Algorithm for Nonlinear Transient Aeroelastic Problems. *AIAA Journal*, Vol. 36, No. 9, pp. 1754-1756.
93. Le Tallec, P. & Mouro, J. (2001). Fluid structure interaction with large structural displacements. *Computer Methods in Applied Mechanics & Engineering* **190** 24-25 3039-3067.
94. Frenklach, M. & Carmer, C. S. 1999. Molecular dynamics using combined quantum & empirical forces: application to surface reactions. *Advances in classical trajectory methods.* Volume 4, 27-63.
95. Haile, J. M. 1992. *Molecular Dynamics Simulations: Elementary Methods*. Wiley.
96. Hase, W. L. 1999. *Molecular Dynamics of Clusters, Surfaces, Liquids, & Interfaces*. *Advances in classical trajectory methods.* Volume 4. JAI Press.
97. Rapaport, D. C. 1995. *The Art of Molecular Dynamics Simulation*. Cambridge University Press.
98. Schlick, T. 2000. *Molecular modeling & simulation. An interdisciplinary guide.* Springer-Verlag, New York.
99. Moelwyn-Hughes, E. A. 1961. *Physical Chemistry*. Pergamon.
100. Terso , J. 1988. Empirical interatomic potential for carbon, with applications to amorphous carbon. *Phys. Rev. Lett.* Vol. 61, pp. 2879-2882.
101. Stillinger, F. H. & Weber, T. A. 1985. Computer simulation of local order in condensed phases of silicon. *Phys. Rev. B* Vol. 31, pp. 5262-5271.