Supplementary Materials for

Three Stacks of Donut HASEL actuator

1. Algorithm for Tilting Control

Three stacks of actuators can be mathematically oriented such that the first actuator stack is aligned with the x axis of the joystick, the second actuator stack is placed at -120 degree away from the first stack, whereas it is 120 degree for the third stack, as shown in Figure 2. The three stacks are represented as unit vectors v1, v2, and v3, respectively on a unit circle whose origin coincides with the origin of the Cartesian coordinates. Similarly, the location of the joystick is represented as a vector highlighted in red, u, with a length varied from 0 to 1 on the same unit circle. The vector length of 0 means that the joystick is not perturbed in any direction, and length of 1 means that the joystick is at the maximum tilting angle.

How much voltage should be applied to any of the actuator stacks depends on the location of the joystick on the unit circle. In order to calculate the amount of voltage needed for each stack, a projection vector of the joystick vector on each actuator’s unit vector is calculated. If and only if the projected vector of a particular stack, v1, v2, or v3, is in the same direction of the unit vector of the same stack, v1, v2, or v3 respectively, which could be checked by evaluating the sign of the two vectors’ dot product, the voltage applied to that particular stack is non-zero and scaled linearly on the length of the projected vector. However, this algorithm design makes the overall Donut stacks to tilt toward the opposite side of the unit circle when compared to the location of the joystick. For instance, moving the joystick toward the first quadrant on the Cartesian coordinates leads to the activation of both the first and the third actuator stacks, making the actuator setup to tilt toward the opposite quadrant, in which the second actuator stack is not activated.

Since the voltage polarity on the Donut actuators must be reversed after every voltage activation, there are two more design constraints needed for the tilting control algorithm. Firstly, all three actuator stacks must be discharged and their voltage polarity must be reversed for the next activation cycle. This constraint could be achieved by constructing a smaller circle concentric to the unit circle. Projected vectors with magnitude less than the radius of the second circle would result in the deactivation and voltage polarity reversal of the corresponding stacks. The radius of this circle could be tuned to account for the imperfection of the joystick’s center. Secondly, as the joystick is rotated 360 degree on the unit circle, each stack must be sequentially activated, deactivated, and discharged, after which the voltage polarity is reversed. The algorithm includes an $if condition$ that as soon as the joystick leaves the active region of a particular stack on the unity circle, which is shown as the negative dot product between the projected vector and the actuator’s unit vector changes from a negative number to a positive number, the stack is discharged and its polarity is reversed.

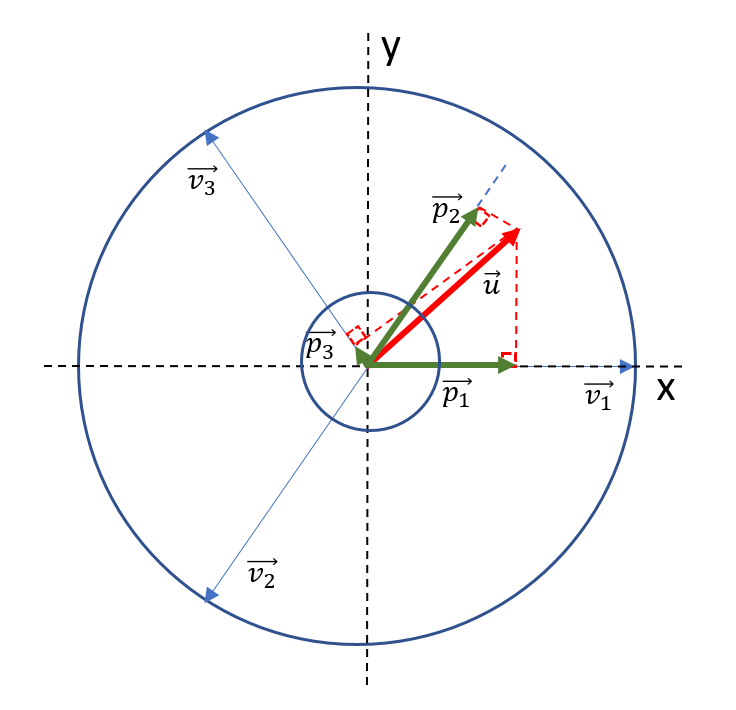


Figure 2: HASEL geometric mapping on Cartesian coordinates

1. Algorithm for variable frequency control

The variable frequency program takes a variable input and uses its value to adjust the frequency of an actuators activation and drain. A potentiometer with a pull-down resistor sends its output signal to the Analog 1 pin of an Arduino Uno. The output signal is then read and stored as a volatile integer sensorValue. Once the sensor value is read, the function actuate is called. The actuate function first determines which polarity the actuator was last actuated at, then activates the actuator in the opposite polarity. This aids in preventing charge build up on the actuator surface. The polarity switch is achieved by switching the pol variable between 1 and 0. The delays are used to adjust the Period of the on/off signal. The Period in milliseconds is defined as 2\*sensorValue and thus the actual operating frequency in Hertz is 500/sensorValue.