Dynamic extension for Ideal Kinematics

Matthew Appleby, Richard Peters

Peters Research Ltd, UK

Abstract. This paper presents a new set of equations for modelling kinematic profiles using a combination of mathematical and computational techniques. The implementation of these equations will extend the capabilities of the kinematic model to produce asymmetric and dynamic profiles. This will provide a more accurate model for standard lift systems as well as enable the modelling of more complex systems.

Keywords: kinematics, lift, elevator, dynamic profile, ideal lift kinematics, twin lift system, 2 cars per shaft, multi, multi-dimensional lift system, simulation

1 INTRODUCTION

Lift kinematics is the study of a lift car in a shaft without reference to mass or force. Variable speed drives can be programmed to match reference velocity profiles. This paper will show a set of equations which can be used to model a lift and produce a kinematic profile which can be used as a reference for lift simulations and physical lifts.

In previous papers regarding lift kinematics, equations have been produced which model symmetric profiles [1] and asymmetric profiles [2]. This paper will show a new set of equations which will model dynamic profiles.

1.1 Symbols

d	Lift journey distance (m)
v_{phase}	Velocity at end of given phase (m/s)
$final\_phase$	Final phase in car journey
a_{phase}	Maximum acceleration (m/s^2)
$j_{phase,0}$	Maximum jerk0 in given phase (rate of change of acceleration) (m/s^3)
$j_{phase,1}$	Maximum jerk1 in given phase (rate of change of acceleration) (m/s^3)
$p_{phase,0}$	Phase start time/period 0 start time
$p_{phase,1}$	Period 1 start time
$p_{phase,2}$	Period 2 start time
$p_{phase,3}$	Period 3 start time
D(t)	Distance travelled at time t (m)
V(t)	Velocity at time t (m/s)
A(t)	Acceleration at time t (m/s^2)
J(t)	Jerk at time t (m/s^3)
d_{min}	Minimum displacement

1.2 Definitions

symmetric profile
the profile of a journey with one target velocity, the same acceleration as deceleration and four identical jerk values. See Fig. 1

asymmetric profile
the profile of a journey with one target velocity but different target acceleration and deceleration or differing jerk values. See Fig. 2

dynamic profile
the profile of a journey with multiple target velocity values. Acceleration and jerk can also vary. See Fig. 3

period
a section of time where the lift is at constant jerk. See Fig. 4 & Fig. 5

phase
a section of time where the lift is changing from one speed to another including the time it remains at its final speed. A period starts and finishes with acceleration of 0 and contains a maximum of four periods. See Fig. 4 & Fig. 6

journey
a section of time where the lift is changing displacement from when the lift begins to move, to when it reaches its destination. Contains a minimum of two phases. See Fig. 4

1.3 Three types of profile

There are three types of kinematic profile that will be referred to in this paper:

1.3.1 Symmetric Profile

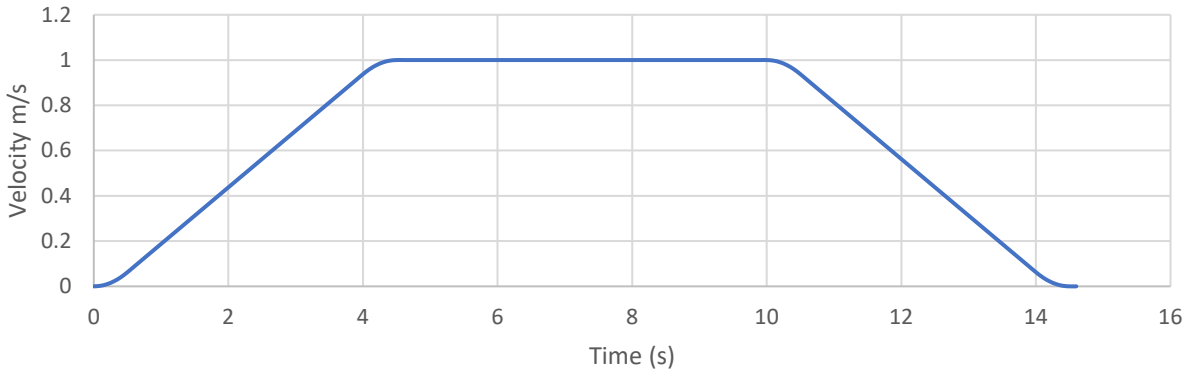


Figure 1 Symmetric Profile

Fig. 1 shows a profile produced when a symmetric model has been used to plot the kinematic profile of a lift. This assumes that there will only be one target velocity and thus the lift is unable to reduce or increase velocity even as the surrounding environment changes. It also makes some assumptions about the kinematic parameters:

$$abs(j_{0,0}) = abs(j_{0,1}) = abs(j_{1,0}) = abs(j_{1,1}) \quad (1)$$

$$abs(a_0) = abs(a_1) \quad (2)$$

$$v_{0,0} = v_{1,1} = v_{final\_phase,1} = 0 \quad (3)$$

This is the most used profile model and the equations for it can be found in 'Ideal Lift Kinematics' by Peters [1] as well as CIBSE Guide D Annex 2 [3]. Most lift systems aim for a symmetric profile to produce a smooth ride quality.

1.3.2 Asymmetric Profile

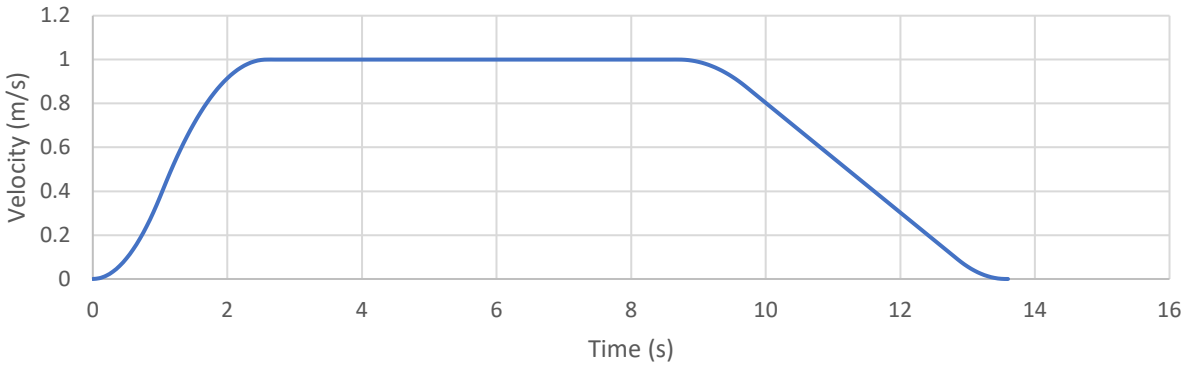


Figure 2 Asymmetric Profile

Fig. 2 shows a profile when an asymmetric model has been used to plot the kinematic profile of a lift. Like the symmetric profile, this model assumes that there will only be one target velocity, however, it allows for different acceleration and deceleration values as well as differing jerk values.

These profiles can be produced using the equations given in 'Quality and quantity of service in lift groups' by Gerstenmeyer [2]. The equations are an extension to the previous ideal lift kinematic equations meaning they can model symmetric and asymmetric equations.

Applications include the modelling of lift cars which have a different acceleration to deceleration, often due to poor motor control. Asymmetric profiles can also be applied to improve lift system performance where there are two cars per shaft as cars sharing a shaft impose additional safety constraints. For example, a lower deceleration may be used when the lower and upper cars need to be moved closer than allowed by the preferred safety distance.

1.3.3 Dynamic Profile

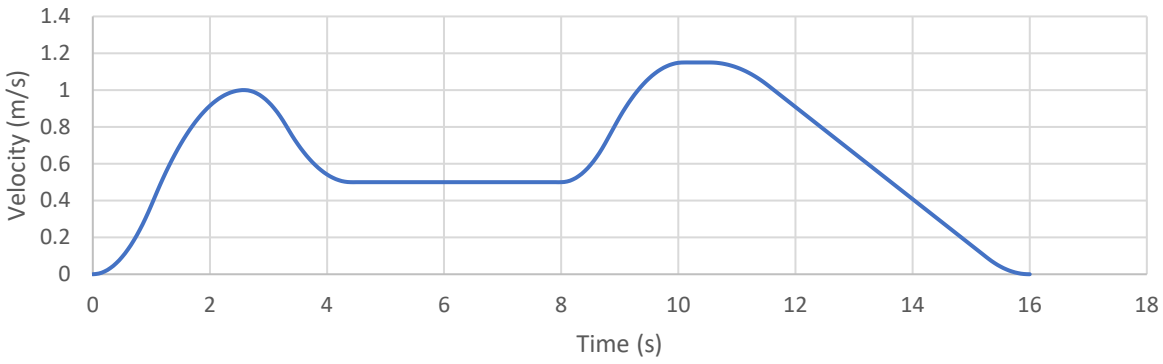


Figure 3 Dynamic Profile

Fig. 3 shows a profile when a dynamic model has been used to plot the kinematic profile of the lift. This level of complexity is achieved when the lift changes its target velocity mid journey to slow down or speed up the car based on the location of the other cars in the shaft. Each update contains a target velocity, jerk0, jerk1, acceleration and a displacement. These update parameters either replace current/future parameters or create a new phase.

This model provides more flexibility than the asymmetric profile, giving more control of the location and motion of the lift car to improve the performance of the system. For example, if a higher car traveling up has been delayed longer than expected, a lower car might need to reduce its velocity until the path is clear for it to accelerate back up to speed. This slow down option makes it more acceptable to start the lower car before the upper car is guaranteed to be moving out of the way. The increase in performance is particularly valuable to multi-dimensional lift systems with more than two lift cars per shaft. [2].

1.4 Three segments of a profile

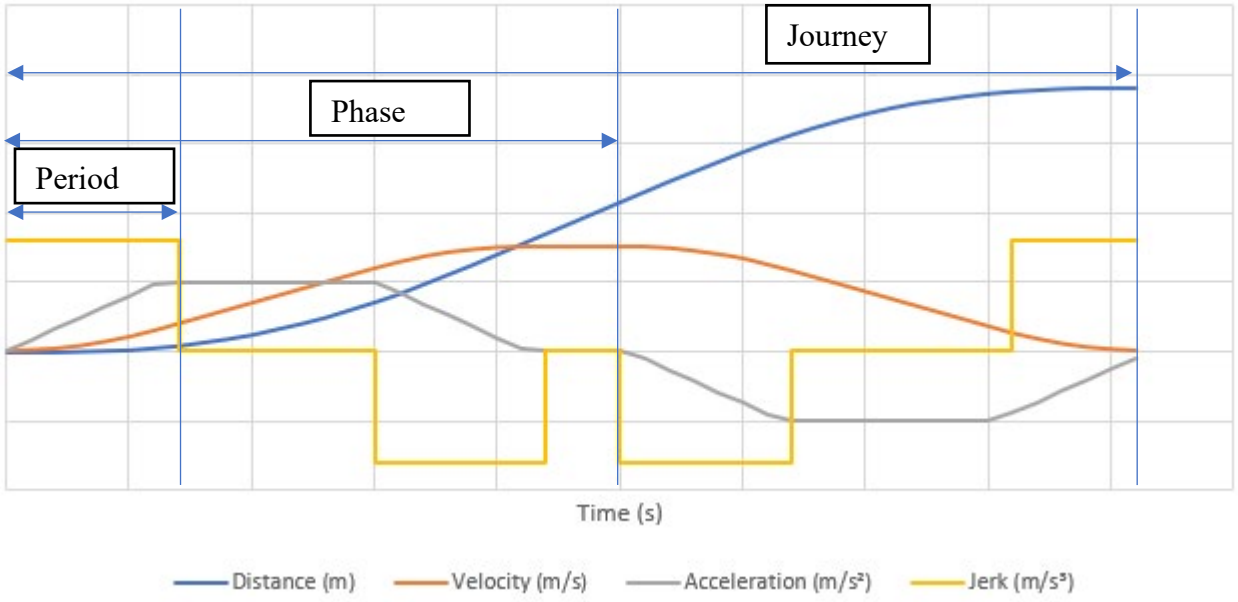


Figure 4 Period, Phase and Profile labelled profile

Fig. 4 shows the kinematic profile of one symmetric journey. The first period and phase, and the journey are labelled.

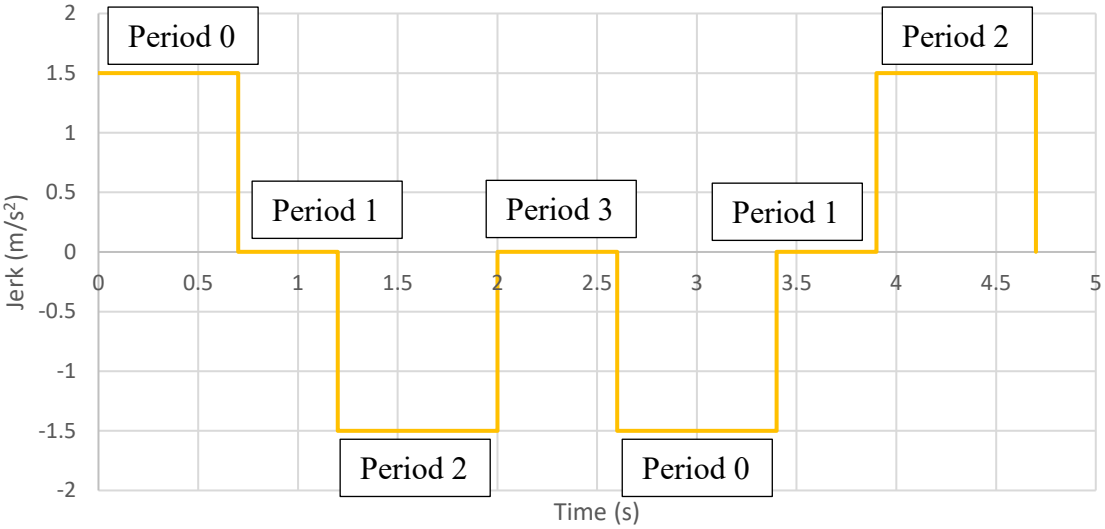


Figure 5 period labelled profile

Fig. 5 shows the jerk profile of one symmetric journey. Each period has a different jerk value and the period count resets to 0 when the car enters a new phase.

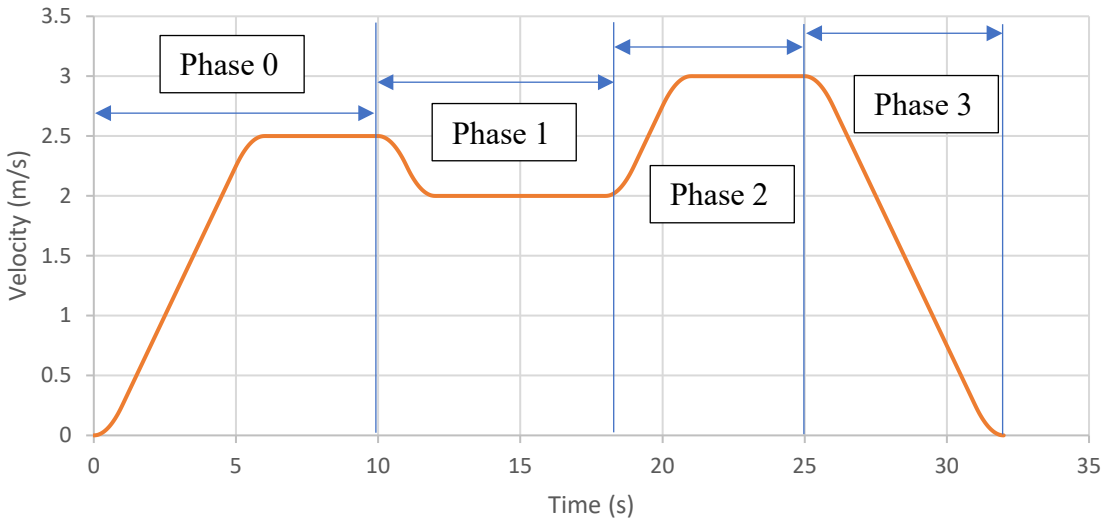


Figure 6 phase labelled profile

Fig. 6 shows the velocity profile of a dynamic profile. The phases have been labelled to demonstrate where phases start and stop on dynamic profiles.

2 OVERVIEW OF PREVIOUS RESEARCH

2.1 Previous Work

2.1.1 Analytical Method

Peters provided a set of equations which model the kinematic profile of a symmetric journey [1]. Each journey is divided into seven periods, each with their own set of equations. Each equation does the entire integration including the addition of the starting value at the previous period. This model provides a straightforward set of individual equations which do not approximate each integration. These equations are transparent and functional but very long and lack flexibility. This is also the method described in Annex 2 of Guide D [3].

Gerstenmeyer provided a set of equations which model the kinematic profile of an asymmetric journey [2]. This uses similar logic to the symmetric model however allows four different jerk inputs and two different acceleration values for the two phases involved. Whilst this improves the flexibility of the model, it also makes the equations even longer and harder to implement.

In the cases where a lift cannot reach the inputted velocity or acceleration, Peters proposed alternative models called ‘case B’ and ‘case C’ [1], Gerstenmeyer however proposed using the same equations by first reducing the velocity and acceleration to the maximum possible values that can be reached [2].

2.1.2 Computational Method

Computational integration methods include quadrature rule, generalised midpoint rule, adaptive algorithms and extrapolation. These methods use calculations which approximate integration to find the profile values without long equations. This method is far more flexible than the analytical method as it does not rely on period separations. However, the approximation required in the computational method decreases the accuracy of the profile. [4]

2.2 Authors’ contribution

The authors have derived an alternative set of equations which map onto the existing equations but allow for more flexible input parameters thus allowing the controller to have more flexibility over the shape of a lift’s kinematic profile. The new equations use a combination of analytical and computational techniques and use the Gerstenmeyer method for dealing with invalid input parameters [2].

3 FUNCTIONAL METHOD

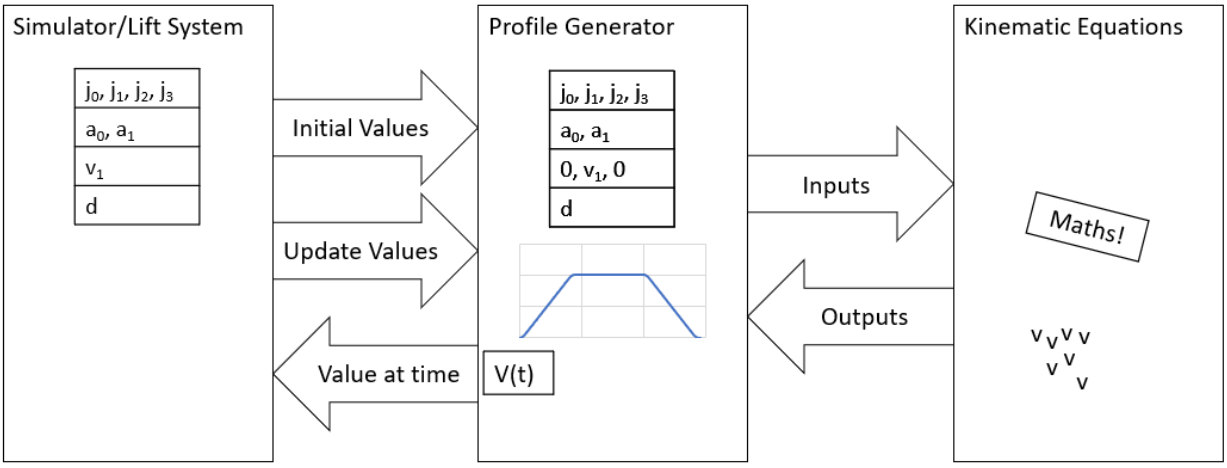


Figure 7 Logic diagram instantiation

The three boxes in Fig. 7 represent the three coded classes. The first box represents the Simulator or Lift System and will be the calling code. The Profile Generator is a class which takes some parameters upon instantiation of an object. The resulting object contains a set of public arrays which plot the profile and can be used by the calling code. The Profile Generator uses the static functions from the Kinematic Equations class to produce the profile. This Kinematic Equations class can be used by other coded classes to perform all the calculations our previous symmetric Kinematic Equations class could do and more.

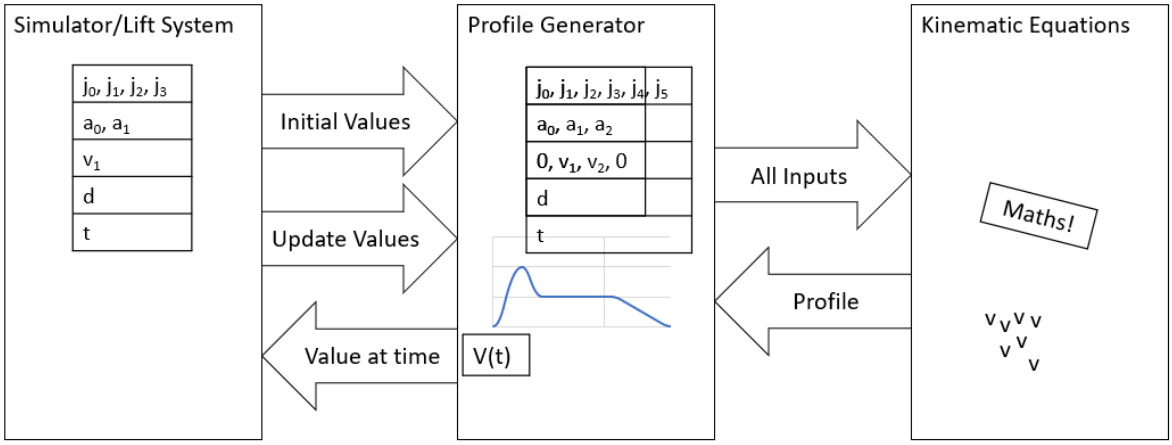


Figure 8 Logic diagram update

For dynamic profiles, the calling code can invoke an update method on the profile object with a new set of inputs including the time at which the update was sent. This will modify the input parameters and then modify the profile arrays.

4 MATHEMATICAL METHOD

4.1 Phase start times

$$p_0 = 0 \tag{4}$$

p_0 will only equal 0 in the first phase. For all the middle phases in a dynamic profile, p_0 will be determined by the time at which the update is sent. For the final phase p_0 is calculated by finding the difference between the target displacement and the minimum displacement and then calculating how long the lift must travel at its final velocity (v_{final_phase}) to reach the target displacement.

$$p_0 = p_3 + \frac{d - d_{min}}{v_{final\_phase}} \tag{5}$$

The period start times $p_{1\text{-}3}$ are calculated using the following equations which can be derived from equations (8-1/2/3) in [2].

$$p_1 = \frac{a_{phase}}{j_{phase[0]}} + p_0 \tag{6}$$

$$p_2 = \frac{v_{phase+1} - v_{phase}}{a_{phase}} - \frac{a_{phase}\left(j_{phase,0} + j_{j_{phase,1}}\right)}{-2 \cdot j_{phase,0} \cdot j_{phase,1}} + p_0 \tag{7}$$

$$p_3 = \frac{v_{phase+1} - v_{phase}}{a_{phase}} + \frac{a_{phase}\left(j_{phase,0} - j_{j_{phase,1}}\right)}{-2 \cdot j_{phase,0} \cdot j_{phase,1}} + p_0 \tag{8}$$

4.2 Kinematic equations

$$J(t) = j \tag{9}$$

Starting with a constant jerk, acceleration can be found by integrating the jerk value with respect to time and adding the starting acceleration at the beginning of the current period (p_{this})

$$A(t) = \int_{p_{this}}^{t} J(time)_{d\ time} + A(p_{this}) \tag{10}$$

Velocity can be calculated two ways, by either adding the velocity change onto the starting velocity (V at p_{this}), or by removing the velocity change from the target velocity (V at p_{next}).

$$V(t) = \int_{p_{this}}^{t} A(time)_{d\ time} + V(p_{this}) \tag{11}$$

$$V(t) = \int_{t}^{p_{next}} A(time)_{d\ time} + V(p_{next}) \tag{12}$$

Displacement is the integration of velocity plus the displacement at the beginning of the current period.

$$D(t) = \int_{p_{this}}^{t} V(time)_{d\ time} + D(p_{this}) \tag{13}$$

See Appendix A for the full equations for each period.

4.3 Allowed Updates

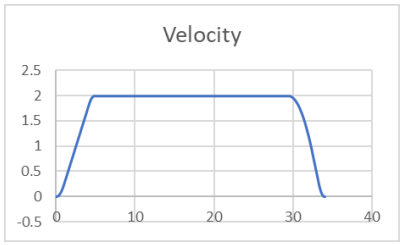


Fig. 9 Initial profile

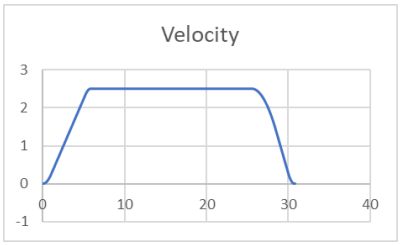


Fig. 10 Period 0 or 1 change

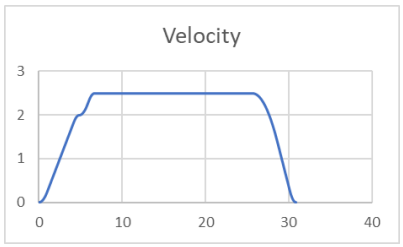


Figure 11 Period 2 change

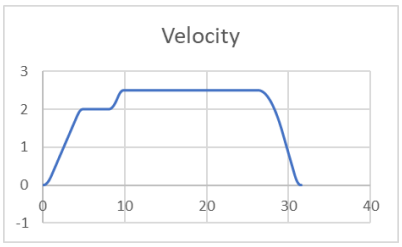


Figure 12 Period 3 change

If the car is in period 0 or 1, a command to increase the velocity can be accepted. The lift will continue accelerating to the higher speed. For example, in Fig. 9 the initial profile was set to reach a velocity of 2m/s. Because the target velocity was changed to 2.5m/s before the end of period 1, the new speed profile is the same as if the higher speed had been commanded at the beginning of the trip, see Fig. 10.

If, however, the lift has entered period 2 and the same command is sent to increase the velocity to 2.5m/s, the lift must finish its current phase before starting to accelerate again up to a higher velocity thus creating a new phase, see Fig. 11. The reason for completing the period is to avoid the added complexities required in changing the jerk mid-period whilst acceleration is not constant.

If the update is sent when the lift is no longer accelerating but is moving at a constant speed in period 3, a new phase is created. The new phase starts at the time the update was sent as seen in Fig. 12.

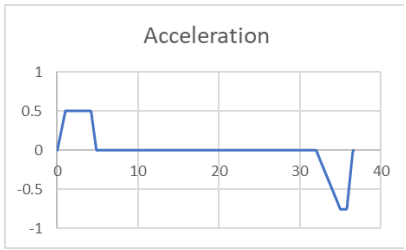


Figure 13 Initial profile

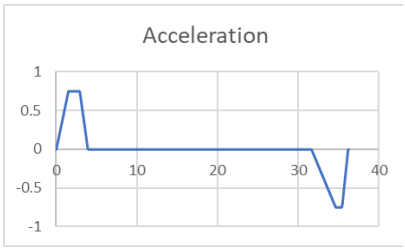


Figure 14 Period 0 change

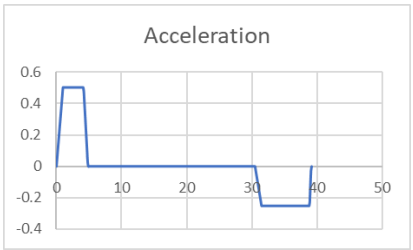


Figure 15 Period 0 - 2 change

If a valid update is sent in period 0 to increase acceleration, the car will continue to jerk up to a higher acceleration. For example, in Fig. 13 the initial profile was set to reach an acceleration of 0.5m/s^2. Because the acceleration was changed to 0.75m/s^2 before the end of period 0, the new

acceleration profile will be the same as if the higher acceleration has been commanded at the beginning of the trip, see Fig. 14.

If a valid update is sent to change the deceleration at any time before the final phase, the working parameters will be updated, see Fig. 15. This update must arrive before the car enters period 3 and, if the deceleration is decreased, before the car would have needed to enter period 3 with the new deceleration.

If an update is sent to increase the displacement and the lift has not entered its final phase, this will always be valid, and the profile should be adjusted accordingly. If the update requests a decreased displacement, it must first be calculated if the lift can stop before the requested displacement in the given jerk, acceleration and velocity.

4.4 Adaptions and rejections

4.4.1 High acceleration

When the car cannot reach the input acceleration and then return to constant velocity without surpassing the target velocity, the acceleration is too high and must be reduced. This is done using equation 8-12 [2].

$$a = \sqrt{\frac{2vj_1j_2}{j_1+j_2}} \tag{14}$$

4.4.2 High velocity

When the car cannot reach the input velocity and then slow down again without surpassing the destination displacement, the velocity is too high and must be reduced.

Due to the use of computational techniques in this method, an analytical equation has not been produced to solve for the maximum possible velocity. Instead, a computational method is used which implements trial and improvement to find the new velocity value.

4.4.3 Low displacement

When the car is mid journey and is sent an update to stop at a displacement lower than the minimum stopping distance, this kinematic update must be rejected.

4.4.4 Late update

When the car is mid journey and is sent an update to change a parameter that has already been used or is impossible to implement, this kinematic update must be rejected. One example of this kind of update is when a command is sent in the final phase of a lift's journey.

5 APPLICATION

5.1 More accurate lift traffic analysis

When modelling lift kinematics, it is currently assumed that the acceleration is the same as the deceleration and that all four jerk values are the same. In a real lift system, acceleration and deceleration can vary in a single journey as seen by accelerometer data. As more data is analysed, more accurate models can be produced when calculating lift system performance.

5.2 Lift systems with two cars per shaft

When two cars share a shaft, the system performance can be improved by giving each car the option to follow a deliberate asymmetric profile thus improving the performance of the system [2].

5.3 Multi-dimensional lift system

For multi-dimensional systems, cars can be given the option to change velocity based on the location of other cars in the system. This can prevent unnecessary stops, improving user experience, and can reduce waiting times, improving performance.

5.4 Improvements to monitoring

Lift performance measurement tools [5] currently try to map the lift's motion to a symmetric profile. Not only will this new model allow monitoring of deliberately asymmetric lifts, data from which will improve simulation inputs, it will also allow for the monitoring of poorly adjusted symmetric lifts thus enhancing maintenance.

6 CONCLUSION

This paper provides a set of equations which accurately model a lift's kinematic profile for symmetric, asymmetric and dynamic journeys. This will enable simulations to model lift motion more accurately as well as improve the performance of lift dispatchers. These improvements will be most noticeable in lift systems with two lift cars per shaft and with multi-dimensional lifts. The equations will also enable improvements in surveying, maintenance and many more technologies.

Currently the solution uses a computational brute force technique to find the maximum velocity and quickest stop floor at any given time slice. There are improvements which can be made to these solutions which will save time and processing power. Although time is less of a concern when simulating a lift system, the need for efficient algorithms becomes more crucial for dispatching and monitoring technologies.

ACKNOWLEDGEMENTS

The authors acknowledge the work of Gabrielle Anderson and her research into the field of lift kinematics with multiple maximum velocities and with separate acceleration and deceleration. The authors would like to thank Nishad Deokar for checking the maths presented in this paper and continuing the research surrounding dynamic kinematics.

7 REFERENCES

[1] R. Peters, "Ideal Lift Kinematics," in *Proceedings of ELEVCON '95*, Hong Kong, 1995.

[2] S. Gerstenmeyer, Quality and quantity of service in lift groups, University of Northampton, 2018.

[3] CIBSE, CIBSE Guide D: Transportation systems in buildings, 2020.

[4] T. Croft, R. Davison Mathematics for engineers, Pearson Prentice Hall, 2015.

[5] R. Peters, “Lift Performance Time,” in *Proceedings of the 2nd Symposium on Lift and Escalator Technologies*, Northampton, 2012.

APPENDIX

Appendix A

Period 0

$$J(t) = j_{phase,0}$$

$$A(t) = j_{phase,0} \cdot t - p_{phase,0}$$

$$V(t) = j_{phase,0} \cdot \frac{(t - p_{phase,0})^2}{2} + v_{phase}$$

$$D(t) = j_{phase,0} \cdot \frac{(t - p_{phase,0})^3}{6} + v_{phase} \cdot (t - p_{phase,0}) + D(p_{phase,0})$$

Period 1

$$J(t) = 0$$

$$A(t) = a_{phase}$$

$$V(t) = a_{phase} \cdot (t - p_{phase,1}) + V(p_{phase,1})$$

$$D(t) = a_{phase} \cdot \frac{(t - p_{phase,1})^2}{2} + V(p_{phase,1}) \cdot (t - p_{phase,1}) + D(p_{phase,1})$$

Period 2

$$J(t) = j_{phase,1}$$

$$A(t) = j_{phase,1} \cdot (t - p_{phase,2}) + a_{phase}$$

$$V(t) = j_{phase,1} \cdot \frac{(t - p_{phase,3})^2}{2} + v_{phase+1}$$

$$D(t) = j_{phase,1} \cdot \frac{(t - p_{phase,3})^3}{6} + v_{phase+1} \cdot (t - p_{phase,2}) - j_{phase,1} \cdot \frac{(p_{phase,2} - p_{phase,3})^3}{6} + D(p_{phase,2})$$

Period 3

$$J(t) = 0$$

$$A(t) = 0$$

$$V(t) = v_{phase,1}$$

$$D(t) = v_{phase,1} \cdot (t - p_{phase,3}) + D(p_{phase,3})$$

Appendix B

Period 0

$$J(t) = j_{phase,0}$$

$$A(t) = \int_{p_{phase,0}}^{t} J(time) \; d\,time$$

$$V(t) = \int_{p_{phase,0}}^{t} A(time) \; d\,time + V(p_{phase,0})$$

$$D(t) = \int_{p_{phase,0}}^{t} V(time) \; d\,time + D(p_{phase,0})$$

Period 1

$$J(t) = 0$$

$$A(t) = a_{phase}$$

$$V(t) = \int_{p_{phase,1}}^{t} A(time) \; d\,time + V(p_{phase,1})$$

$$D(t) = \int_{p_{phase,1}}^{t} V(time) \; d\,time + D(p_{phase,1})$$

Period 2

$$J(t) = j_{phase,2}$$

$$A(t) = \int_{p_{phase,2}}^{t} J(time) \; d\,time + A(p_{phase,2})$$

$$V(t) = \int_{t}^{p_{phase,3}} A(time) \; d\,time + V(p_{phase,3})$$

$$D(t) = \int_{p_{phase,2}}^{t} V(time) \; d\,time + D(p_{phase,2})$$

Period 3

$$J(t) = 0$$

$$A(t) = 0$$

$$V(t) = v_{phase,1}$$

$$D(t) = \int_{p_{phase,3}}^{t} V(time) \; d\,time + D(p_{phase,3})$$

Appendix C

Function

$$Acurve(t, j, a) = j \cdot t + a$$

$$Vcurve(t, j, v) = \frac{j}{2} \cdot t^2 + v$$

$$Vdiagonal(t, a) = a \cdot t$$

Period 0

$$J(t) = j_{phase,0}$$

$$A(t) = Acurve((t - p_{phase,0}), (j_{phase,0}), 0)$$

$$V(t) = Vcurve\big((t - p_{phase,0}), j_{phase,0}, v_{phase}\big)$$

$$D(t) = j_{phase,0} \cdot \frac{\big(t - p_{phase,0}\big)^3}{6} + v_{phase} \cdot (t - p_{phase,0}) + D(p_{phase,0})$$

Period 1

$$J(t) = 0$$

$$A(t) = a_{phase}$$

$$V(t) = Vdiagonal\big((t - p_{phase,1}), a_{phase}\big) + V(p_{phase,1})$$

$$D(t) = a_{phase} \cdot \frac{(t - p_{phase,1})^2}{2} + V(p_{phase,1}) \cdot (t - p_{phase,1}) + D(p_{phase,1})$$

Period 2

$$J(t) = j_{phase,1}$$

$$A(t) = Acurve((t - p_{phase,2}), (j_{phase,1}), (a_{phase}))$$

$$V(t) = Vcurve\big((t - p_{phase,3}), j_{phase,1}, v_{phase+1}\big)$$

$$D(t) = j_{phase,1} \cdot \frac{(t - p_{phase,3})^3}{6} - j_{phase,1} \cdot \frac{(p_{phase,2} - p_{phase,3})^3}{6} + v_{phase+1} \cdot (t - p_{phase,2}) + D(p_{phase,2})$$

Period 3

$$J(t) = 0$$

$$A(t) = 0$$

$$V(t) = v_{phase,1}$$

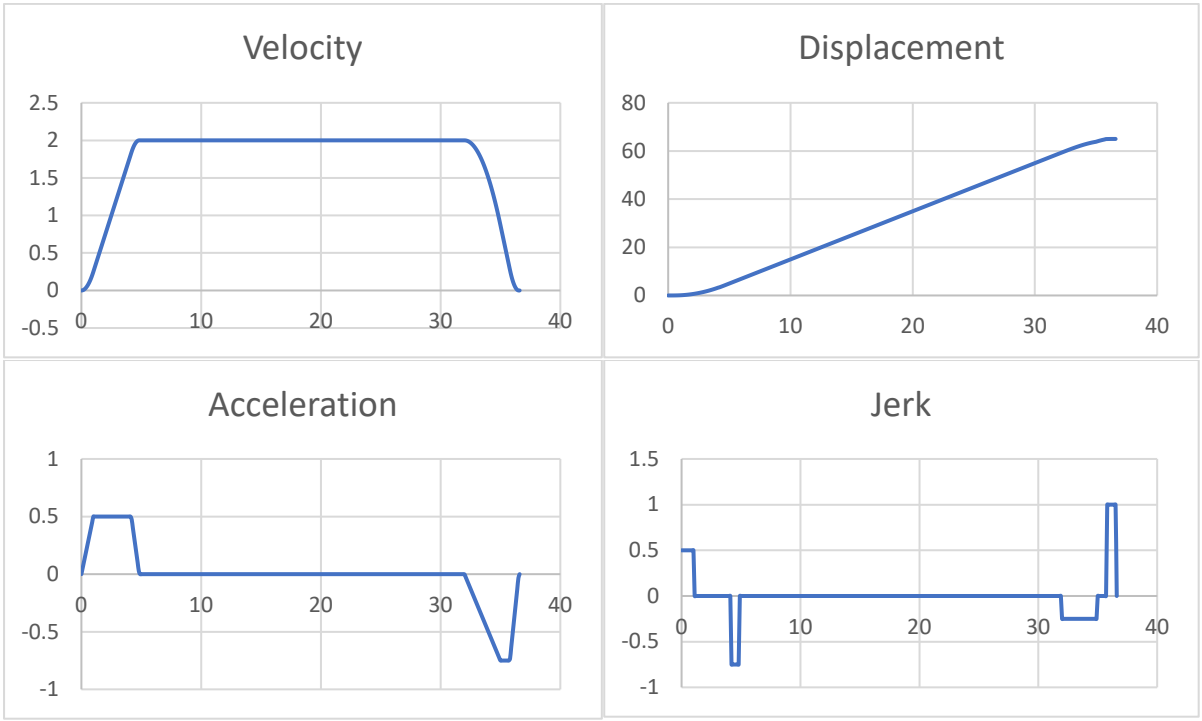
$$D(t) = v_{phase,1} \cdot (t - p_{phase,3}) + D(p_{phase,3})$$

Appendix D

The following appendix contains examples of input parameters and resulting profiles. They attempt to demonstrate all categories of profile and can be used as test cases for any future developments.

Initiate

```
jerk[4] = { 0.5, 0.75, 0.25, 1.0 };
acceleration[2] = { 0.5, 0.75 };
velocity = 2.0;
displacement = 65.0;
```

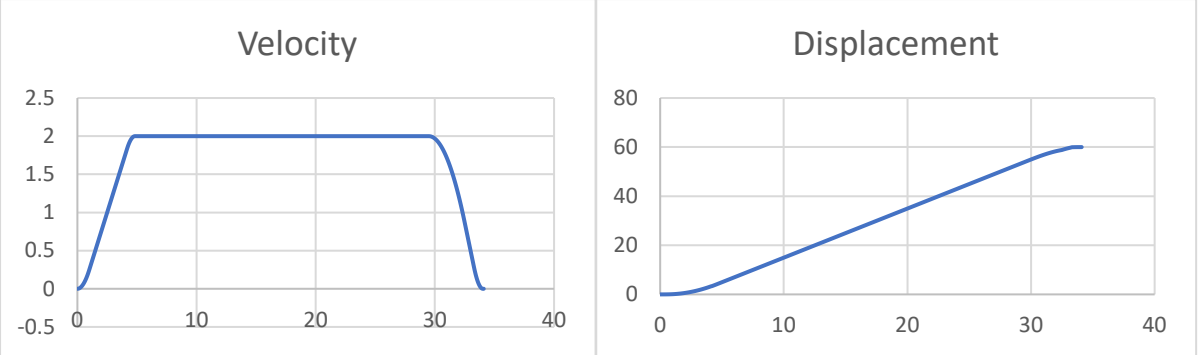


This is an asymmetric kinematic profile generated by the starting variables. In many cases, this will be enough to plot the lift journey.

Period 1 - 4 displacement change

At t = 1.5, go to displacement = 60

```
if (running_time > 1.49 && running_time < 1.51) {
       displacement = 60.0;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

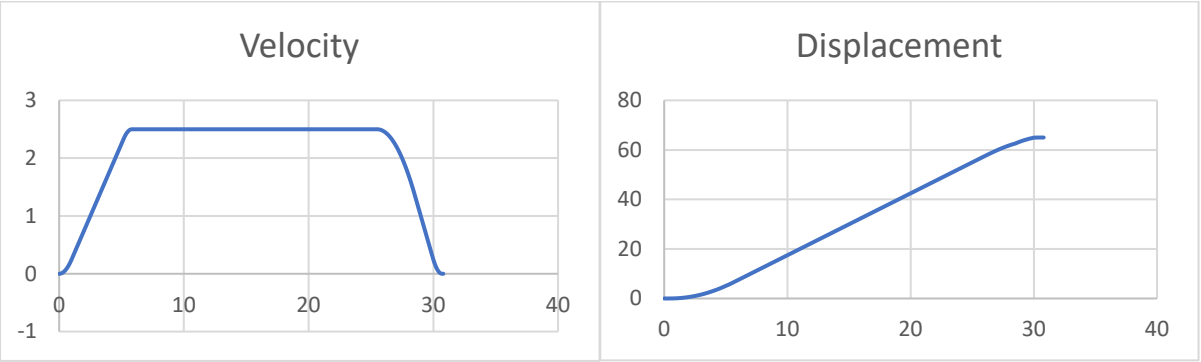


In this case, up to the time of change, the profile for displacement = 60.0 and displacement = 65.0 is the same. This means the code can delete the old displacement target and adopt the new one. This is an asymmetric profile. The result is the same as if the initiator had been given 60.0 instead of 65.0.

Period 1 & 2 velocity change

At t = 1.5, go to velocity = 2.5

```
if (running_time > 1.49 && running_time < 1.51) {
       velocity = 2.5;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```



In this case, up to the time of change, the profile for velocity = 2.5 and velocity = 2.0 is the same. This means the code can delete the old velocity target and adopt the new one. This is an asymmetric profile. The result is the same as if the initiator had been given 2.5 instead of 2.0.

Period 3 velocity change

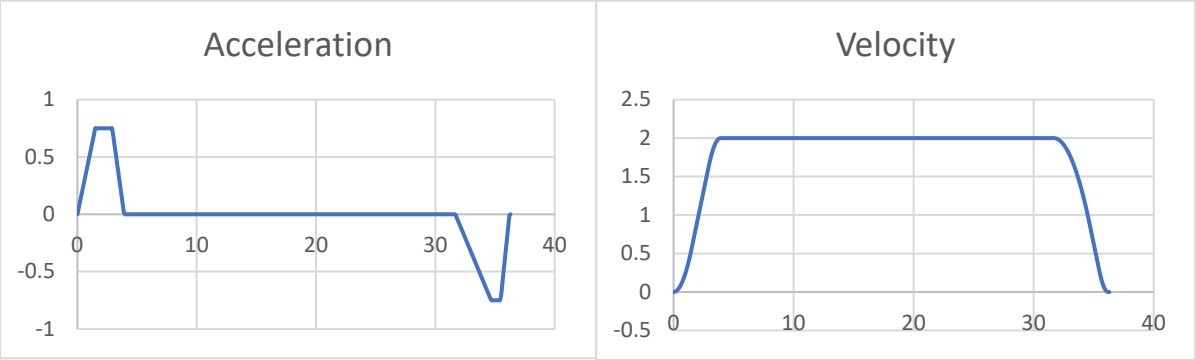
At t = 4.5, go to velocity = 2.5

```
if (running_time > 4.49 && running_time < 4.51) {
       velocity = 2.5;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

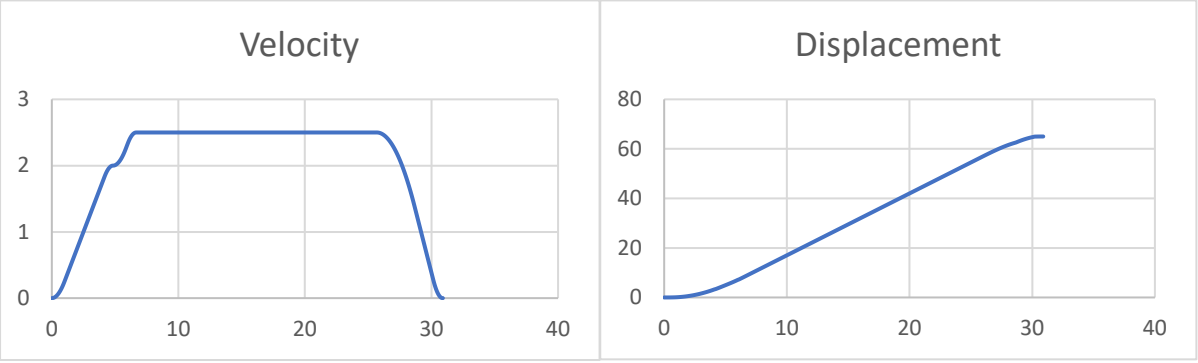
Period 1 acceleration change

At t = 0.5, go to acceleration = 0.75

```
if (running_time > 0.49 && running_time < 0.51) {
       acceleration[0] = 0.75;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```



In this case, up to the time of change, the profile for acceleration[0] = 0.75 and acceleration[0] = 0.5 is the same. This means the code can delete the old acceleration target and adopt the new one. This is an asymmetric profile. The result is the same as if the initiator had been given 0.75 instead of 0.5.



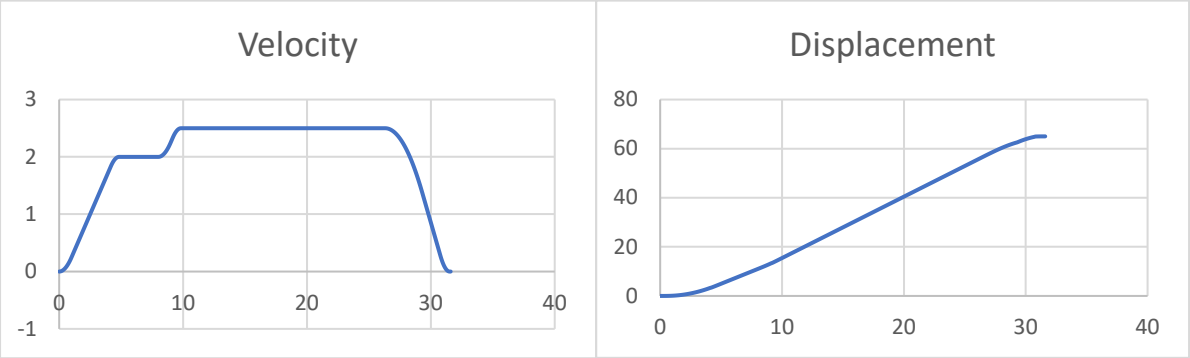
In this case, the change in velocity is buffered and only implemented once period 3 has completed. Halting period 3 part way and returning to period 2 presented a great challenge. This is a dynamic profile. The profile is equivalent to the change being made at t = 4.9.

This could be improved in future versions.

Period 4 velocity change

At t = 8, go to velocity = 2.5

```
if (running_time > 7.99 && running_time < 8.01) {
       velocity = 2.5;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

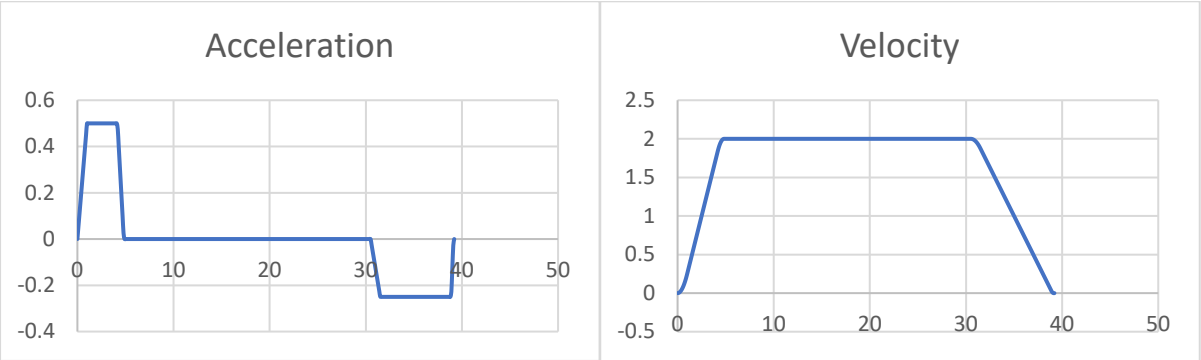


At t = 8, the lift begins accelerating to its new velocity. This will likely be the most common type of dynamic profile change.

Period 1 - 4 deceleration change

At t = 1.5, go to deceleration = `0.25`

```
if (running_time > 1.49 && running_time < 1.51) {
       acceleration[1] = 0.25;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

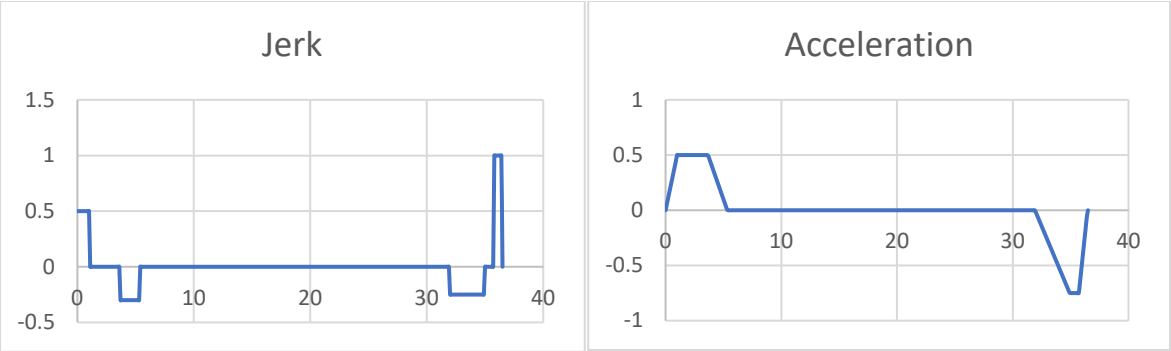


In this case, up to the time of change, the profile for acceleration[1] = 0.75 and acceleration[1] = 0.25 is the same. This means the code can delete the old acceleration target and adopt the new one. This is an asymmetric profile. The result is the same as if the initiator had been given 0.25 instead of 0.75.

Period 1 & 2 jerk 2 change

At t = 1.5, go to jerk 2 = 0.5

```
if (running_time > 1.49 && running_time < 1.51) {
       jerk[1] = 0.5;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```



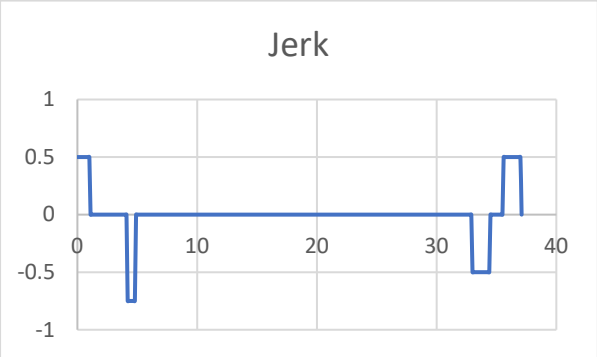
In this case, up to the time of change, the profile for jerk[1] = 0.5 and jerk[1] = 0.75 is the same. This means the code can delete the old acceleration target and adopt the new one. This is an asymmetric profile. The result is the same as if the initiator had been given 0.5 instead of 0.75.

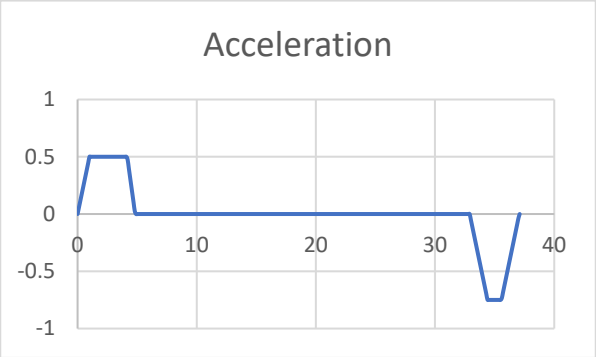
Period 1 - 4 jerk 3 & 4 change

At t = 1.5, go to jerk 3 = 0.5 and jerk 4 = 0.5

```
if (running_time > 1.49 && running_time < 1.51) {
       jerk[2] = 0.5;
       jerk[3] = 0.5;
       profile_one.ChangeProfile(running_time,
```

```
                                jerk,
                                acceleration,
                                velocity,
                                displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```



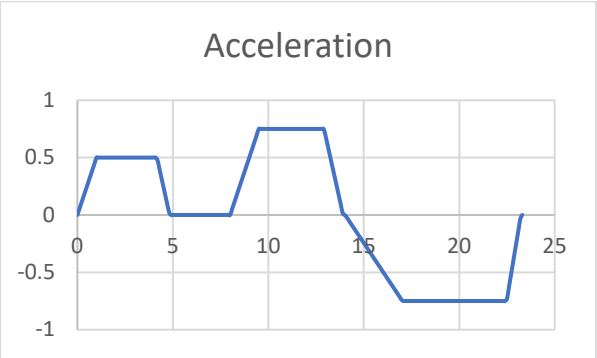


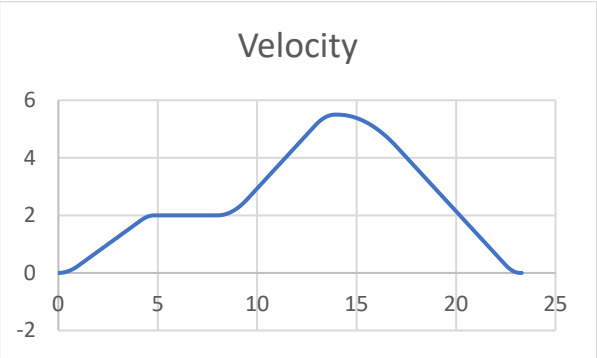
Modified inputs

Too high velocity change

At t = 8, go to velocity = 10

```
if (running_time > 7.99 && running_time < 8.01) {
        velocity = 10.0;
        profile_one.ChangeProfile(running_time,
                                jerk,
                                acceleration,
                                velocity,
                                displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```





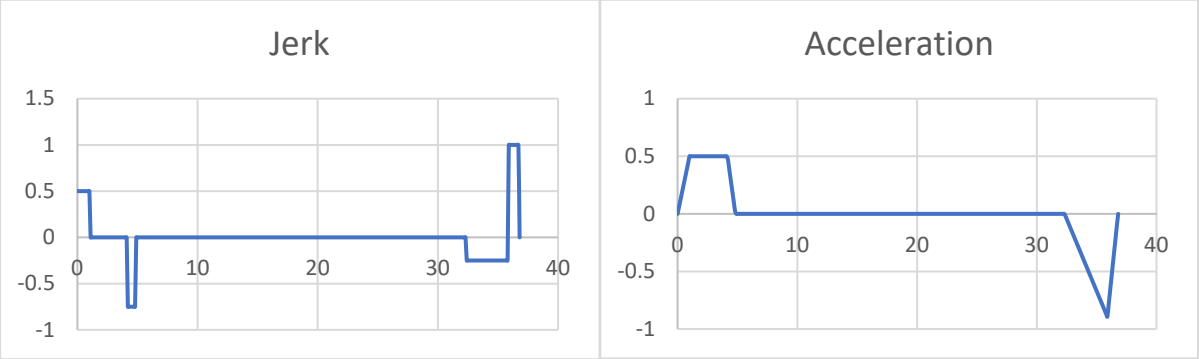
The velocity has been reduced to 5.5 which is the maximum velocity value, to one decimal place, that will allow the lift to stop before passing the destination floor.

Too high acceleration change

At t = 8, go to deceleration = 1.0

```
if (running_time > 1.49 && running_time < 1.51) {
        acceleration[1] = 1.0;
```

```
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```



The acceleration value has been reduced to allow the lift to stop before passing the destination floor. The input was 1.0 but the lift never exceeds 0.89 in the final decelerating period.

Rejected Changes

Too short displacement change

At t = 8, go to displacement = 10

```
if (running_time > 7.99 && running_time < 8.01) {
        displacement = 10.0;
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
         time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift will not be able to stop in time to meet a displacement of 10m

Final phase displacement change

At t = 35, go to displacement = 60

```
if (running_time > 34.99 && running_time < 35.01) {
        displacement = 60.0;
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
         time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift has already passed 60m displacement

Final phase velocity change

At t = 35, go to velocity = 2.5

```
if (running_time > 34.99 && running_time < 35.01) {
       velocity = 2.5;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift is currently decelerating so it is too late to change the velocity.

Period 2 - 4 acceleration change

At t = 1.5, go to acceleration = 0.25

```
if (running_time > 34.99 && running_time < 35.01) {
       acceleration[0] = 0.25;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift is currently accelerating or has finished accelerating so it is too late to change the acceleration.

Final phase deceleration change

At t = 35, go to deceleration = 0.25

```
if (running_time > 34.99 && running_time < 35.01) {
       acceleration[1] = 0.25;
       profile_one.ChangeProfile(running_time,
                                 jerk,
                                 acceleration,
                                 velocity,
                                 displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift is currently decelerating so it is too late to change the deceleration.

Period 1 - 4 jerk 1 change

At t = 1.5, go to jerk 1 = 0.25 and jerk 2 = 0.25

```
if (running_time > 1.49 && running_time < 1.51) {
```
jerk[0] = 0.25;

```
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift is currently accelerating or has finished accelerating so it is too late to change the jerk 1 or 2.

Period 3 & 4 jerk 2 change

At t = 8, go to jerk 2 = 0.3

```
if (running_time > 7.99 && running_time < 8.01) {
        jerk[1] = 0.5;
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift has already reached of passed the jerk 2 period

Final phase jerk 3 & 4 change

At t = 35, go to jerk 3 = 0.5 and jerk 4 = 0.5

if (running_time > 34.99 && running_time < 35.01) {

jerk[2] = 0.5;

```
        jerk[3] = 0.5;
        profile_one.ChangeProfile(running_time,
                                  jerk,
                                  acceleration,
                                  velocity,
                                  displacement);
        time_of_destination = profile_one.TimeOfDestination();
}
```

This change is rejected as the lift is currently decelerating so it is too late to change the jerk values.