PaceGuide

Intelligent Race Pacing via GPS-Based Control Systems

Hocine Filali
Systems Engineering and Operations Research
George Mason University
Fairfax, VA
hfilali@gmu.edu

Amr Hamza
Systems Engineering and Operations Research
George Mason University
Fairfax, VA
ahamza9@gmu.edu

Adewale Adekambi
Systems Engineering and Operations Research
George Mason University
Fairfax, VA
aadekamb@gmu.edu

***Abstract*— This paper presents the development and analysis of an automated race pace control system that helps runners achieve target race completion times. We developed four progressive control models, starting with basic proportional speed control and culminating in a system that combines both velocity and position feedback for more accurate pacing. Our analysis investigates control gain effects ($K = 0.1, 1, 6$), measurement intervals ($\Delta t = 1s, 4s, 10s$), GPS position error ($\sigma^2 = 2m$), and combined velocity-position control (Kp, Ki). Initial models demonstrated that proportional control with $K = 1$ achieved reasonable speed tracking with minimal position shortfall (3m), while excessive gains ($K = 6$) led to a quicker response. The final integrated model incorporating both velocity and position feedback showed better performance in maintaining target pace while correcting small errors in position over time.**

I. INTRODUCTION

The challenge of maintaining a consistent pace during long-distance running has traditionally been a significant barrier to achieving target race times. This challenge was prominently seen during Eliud Kipchoge's sub-2-hour marathon attempt, where a pace car helped him maintain a steady speed. While elite athletes benefit from such sophisticated pacing systems, recreational runners typically rely on periodic watch checks and mental calculations, leading to poor pacing strategies. Our research bridges this gap by developing a GPS-based smart watch control system that provides real-time pace feedback. The system processes position data to compute velocity and compares it against target speeds, providing simple visual feedback (faster/slower/maintain) to the runner. We approached this challenge through four iterative models:

1. A basic proportional control system using instantaneous velocity feedback.

2. A realistic model incorporating discrete-time velocity measurements.

3. An enhanced system accounting for GPS measurement errors.

4. A comprehensive dual-input controller using both velocity and position feedback.

Each model builds upon the previous one, addressing real-world complications such as measurement intervals, (Δt effects ranging from 1 to 10 seconds), GPS position uncertainty ($\sigma^2 =$ 2m), and the need for cumulative position error correction. This systematic approach allows us to understand how each factor affects control performance and develop robust strategies for maintaining target pace despite sensor limitations.

The significance of this work extends beyond competitive running, this can also be related to the broader field of human-in-the-loop control systems where precise speed regulation/ calculation must be maintained despite noise and variable human responses.

II. MODEL 1 – PROPORTIONAL CONTROL

A. *Model Strurcture :*

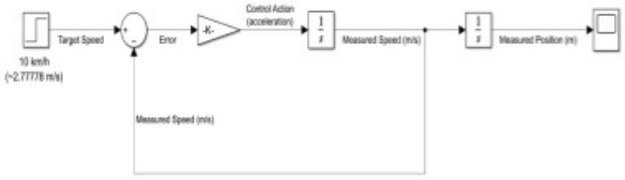

Figure 1. Proportional Control Model Block Diagram

Model Goal

The goal of this model is to design and analyze a proportional control system that ensures a runner maintains a target speed and achieves a desired position of 10 kilometers at a specific time of 1 hour. The runner adjusts their acceleration proportionally based on the error between the target speed and the current measured speed.

Model Description:

The system's input is the *target speed* (e.g., 10 km/h ≈ 2.77778 m/s). The *measured speed* of the runner is compared to the target speed to calculate the error. A proportional control action is implemented: the runner's acceleration is proportional to this error. The proportional gain K dictates the response magnitude. The system includes integrators ($\frac{1}{s}$) to model the following relationships:

- Acceleration integrating into speed.

- Speed integrating into position.

The model uses a proportional controller to calculate the control action (acceleration). The speed and position are computed using successive integration of the control action over time. Finally, the *actual position* over time is plotted. The model will analyze how well the runner achieves the target in 1 hour.

Equations:

Velocity Equation:

When solving for velocity in this context, the equation is: $v(t) = (target\ speed) - (target\ speed)e^{-kt}$. Looking at the equation, the control action (K) determines how quickly the runner accelerates and reaches full speed (steady state). As K increases, the time-constant decreases.

Error Calculation:

The error is calculated as $e(t) = v_{target} - v_{measured}$, where *e(t) (m/s)* is the current speed error, v_{target} *(m/s)* is the target speed, *and* $v_{measured}$ *(m/s)* is the measured speed at time *t*.

Proportional Control Law

The control action is calculated as $a(t)=K \cdot e(t)$, where a(t) *(m/s^2)* is the acceleration at time *t* and *K* is the proportional gain.

Model Analysis

When K=1:

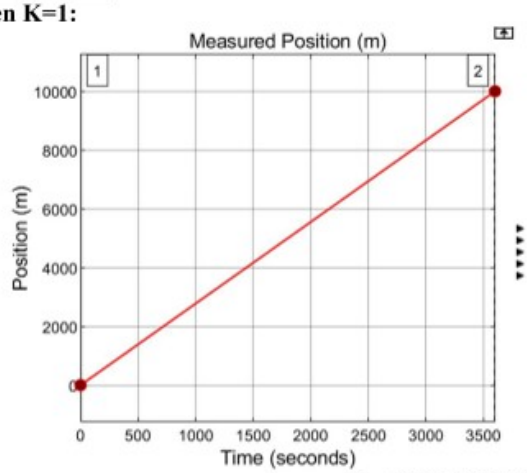

Figure 2. Model 1 K=1 Graph

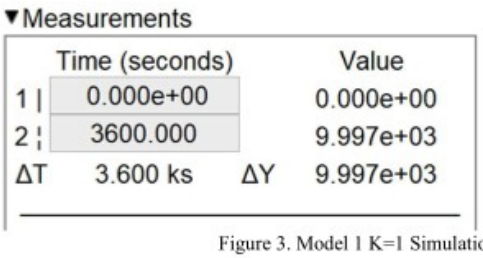

Figure 3. Model 1 K=1 Simulation Output

Rationale:

When K is equal to 1, the runner fails to meet the objective of finishing the race (10km in 1 hour) in time and falls short by 3 meters.

When K=6:

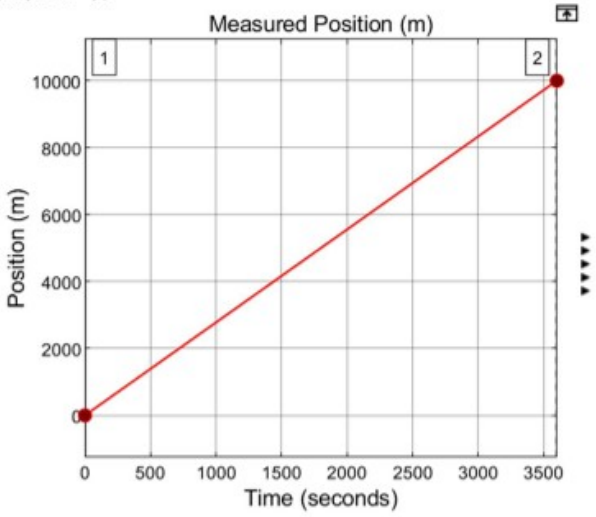

Figure 4. Model 1 K=6 Graph

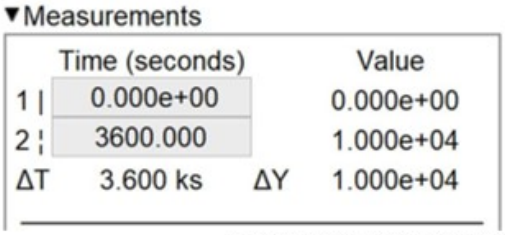

Figure 5. Model 1 K=6 Simulation Output

Rationale:

When K is equal to 6, the runner is able to meet the objective and complete the race in time.

When K=0.1:

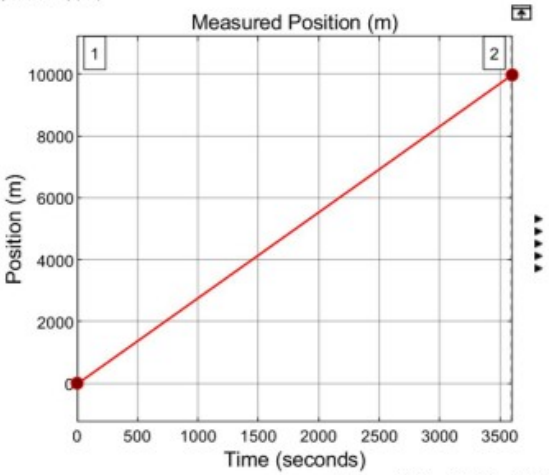

Figure 6. Model 1 K=0.1 Graph

▼Measurements

	Time (seconds)		Value
1	0.000e+00		0.000e+00
2	3600.000		9.972e+03
ΔT	3.600 ks	ΔY	9.972e+03

Figure 7. Model 1 K=0.1 Simulation Output

Rationale:

When K is equal to 0.1, the runner fails to meet the objective, again. This time, the runner falls short by 28 meters.

Conclusion

The proportional control model provides an effective mechanism to regulate the runner's speed and position. By tuning the control gain *K*, the system can be optimized to balance responsiveness and stability, enabling the runner to meet the target. Comparing different gain values highlights the measured distance of the runner after 1 hour to see if it matches the objective.

III. Model 2 – Proportional Control With Time Delay

A. Model Strurcture :

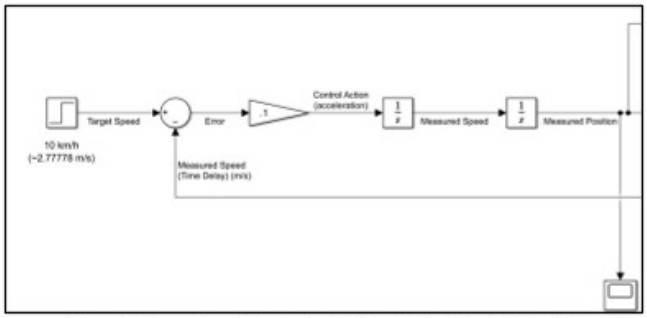

Figure 8. Proportional Control with Time Delay Model Block Diagram

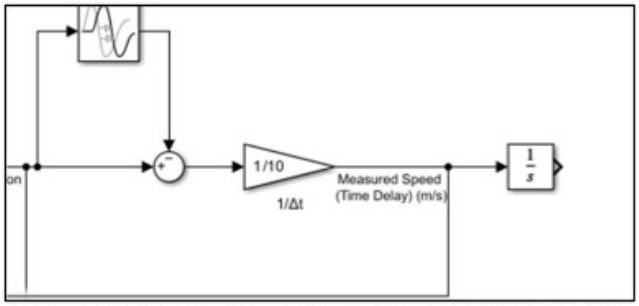

Figure 9. Proportional Control Model with Time Delay Block Diagram Cont.

Model goal

The goal is to evaluate the effectiveness of a control system in enabling a runner to achieve a target velocity. The model computes velocity based on position measurements taken at discrete time intervals Δt instead of instantaneous velocity measurements, reflecting real-world constraints. Different values of t are tested to assess their impact on the system's performance.

Equations

Error Calculation:

Error e(t) is calculated as: $e(t)=v_{target}-v_{measured}$. A proportional controller ($K = 0.1$) generates the control action: $a(t)=K \cdot e(t)$.

Kinematic Model:

Speed $v(t)$ and position $x(t)$ are related through the integrators:

$$v(t) = \int a(t)dt$$

$$x(t) = \int v(t)dt$$

Measured Velocity Calculation:

The measured velocity is computed as:

$$v_{measured}(t) = \frac{x(t) - x(t - \Delta t)}{\Delta t}$$

Feedback Loop:

The measured velocity $v_{measured}(t)$ is fed back to compare with the target velocity $v_{target}(t)$ allowing the controller to adjust acceleration dynamically.

Model Analysis

When: Δt =10:

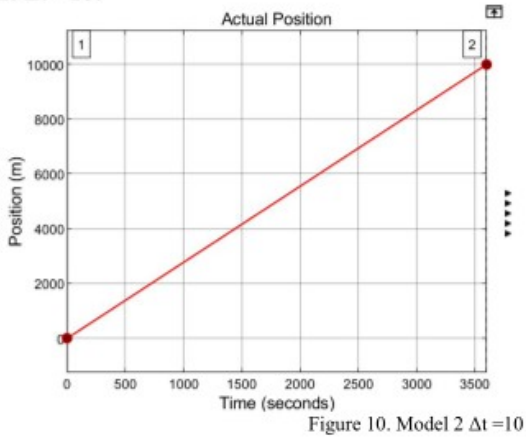

Figure 10. Model 2 Δt =10 Graph

▼Measurements

	Time (seconds)		Value
1	0.000e+00		0.000e+00
2	3600.000		9.987e+03
ΔT	3.600 ks	ΔY	9.987e+03

Figure 11. Model 2 Δt =10 Simulation Output

Rationale:

When **Δt** is equal to 10, the runner fails to meet the objective of finishing the race (10km in 1 hour) in time and falls short by 13 meters.

When Δt=4:

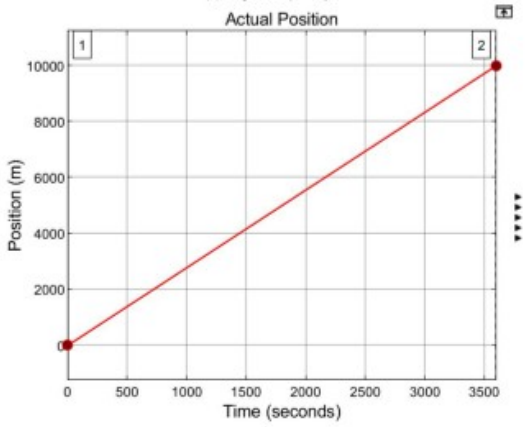

Figure 12. Model 2 Δt =4 Graph

▼Measurements

	Time (seconds)		Value
1	0.000e+00		0.000e+00
2	3600.000		9.977e+03
ΔT	3.600 ks	ΔY	9.977e+03

Figure 13. Model 2 Δt =4 Simulation Output

Rationale:

When **Δt** is equal to 4, the runner fails to meet the objective of finishing the race (10km in 1 hour) in time and falls short by 23 meters.

When Δt=1:

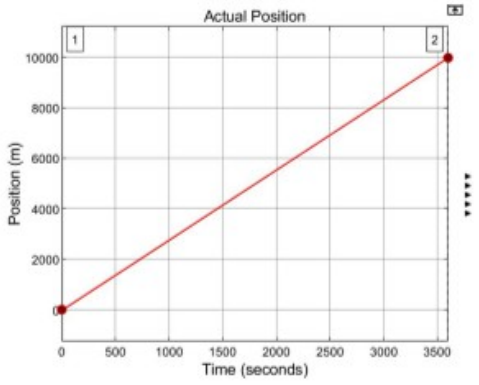

Figure 14. Model 2 Δt =1 Graph

▼Measurements

	Time (seconds)		Value
1	0.000e+00		0.000e+00
2	3600.000		9.971e+03
ΔT	3.600 ks	ΔY	9.971e+03

Figure 15. Model 2 Δt =1 Simulation Output

Rationale:

When **Δt** is equal to 1, the runner fails to meet the objective of finishing the race (10km in 1 hour) in time and falls short by 29 meters.

IV. MODEL 3 – PROPORTIONAL CONTROL, TIME DELAY, RANDOM ERROR

MODEL STRUCTURE

The third model builds upon our previous iterations by introducing random measurement errors to simulate real-world GPS inaccuracies in position tracking. This addition creates a more realistic simulation of actual running conditions where all position measurements contain independent noise components.

Just as in Model 2, the system computes velocity by measuring position changes over time intervals. However, Model 3 adds independent random error components ($\varepsilon 1$ and $\varepsilon 2$) to both the current and delayed position measurements before calculating velocity. This better reflects how actual GPS devices work, where atmospheric conditions, satellite geometry, and other factors introduce measurement uncertainties that affect each position reading independently. Since both the current and previous position measurements are obtained from a GPS, each contains its own measurement error, leading to compounded uncertainty when calculating velocity.

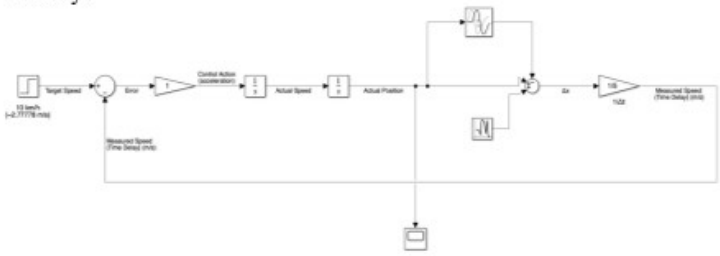

Figure 16. Random Error Model Block Diagram

MATHEMATICAL FRAMEWORK

The core equations remain similar to Model 2, but with the addition of random error.

The measured velocity is calculated as:

$$v_{measured} = \left(\left(x_{final} + \varepsilon_1 \right) - \left(x_{initial} + \varepsilon_2 \right) \right) / \Delta t$$

Where:
x_{final} represents the current position

$x_{initial}$ represents the position after the time delay

ε represents the random error term from a normal distribution $N(\mu=0, \sigma^2=var)$

Δt represents the time delay interval (1 second)

In control theory terms, we introduce a **stochastic** element to our feedback loop. The error term ε follows a normal distribution with the following parameters:

- Mean (μ) = 0 (unbiased measurements)
- Variance (σ^2) = {2, 20, 200} (three test cases)
- Random seed = 0 (for reproducibility)
- Sample time = 1 second

The control law remains unchanged from Model 2:

$$a(t) = K \cdot e(t)$$

Where:
a(t) is the acceleration command
K is the proportional gain
e(t) is the velocity error *(vtarget - vmeasured)*

MODEL ANALYSIS

Case 1: Low Noise (Variance = 2)

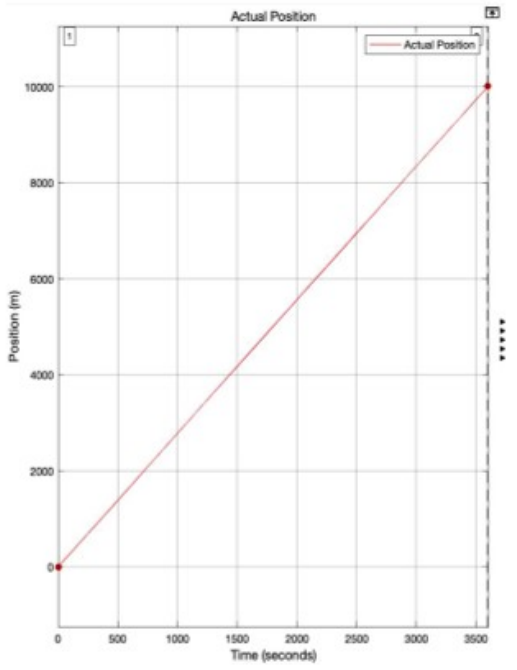

Figure 17. Model 3 $\sigma^2 = 2$ Graph

▼ Measurements

	Time (seconds)		Value
1	0.000e+00		0.000e+00
2	3600.000		1.001e+04
ΔT	3.600 ks	ΔY	1.001e+04

Figure 18. Model 3 $\sigma^2 = 2$ Simulation Output

With minimal variance ($\sigma^2 = 2$), the system demonstrates excellent stability and tracking performance. The position graph shows:

- Small, high-frequency oscillations around the ideal trajectory
- Minimal deviation from the target path
- Quick recovery from measurement disturbances
- Overall position tracking accuracy within $\pm 2\sqrt{2}$ meters

This case most closely resembles the behavior of Models 1 and 2, suggesting that low levels of measurement noise don't significantly impact system performance.

Case 2: Medium Noise (Variance = 20)

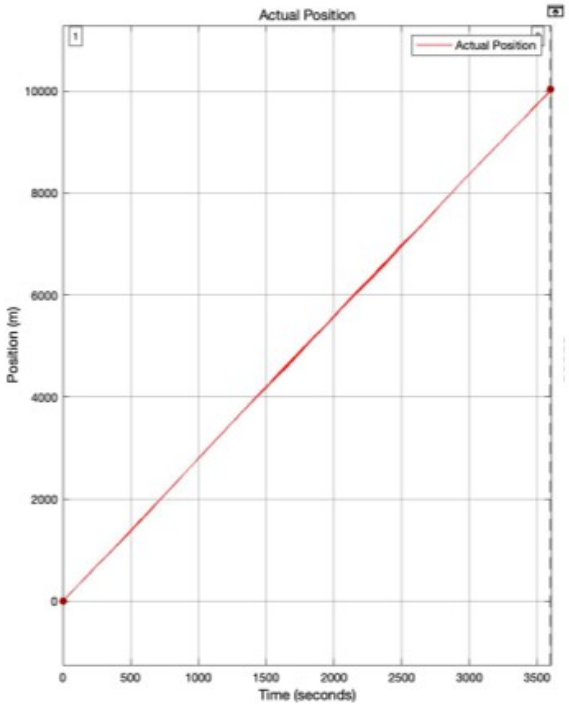

Figure 19. Model 3 $\sigma^2 = 20$ Graph

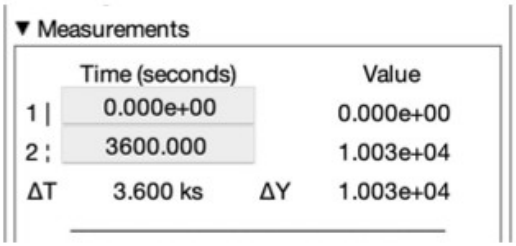

Figure 20. Model 3 $\sigma^2 = 20$ Simulation Output

Increasing the variance to $\sigma^2 = 20$ reveals more pronounced effects:

- Visible oscillations in the position trajectory
- Larger deviations from the ideal path
- Control system actively compensating for errors
- Position tracking accuracy within $\pm 5\sqrt{2}$ meters
- System maintains overall stability despite increased noise

This represents a more realistic scenario for consumer-grade GPS devices, where measurement errors are noticeable but not severe enough to compromise the system's fundamental functionality.

Case 3: High Noise (Variance = 200)

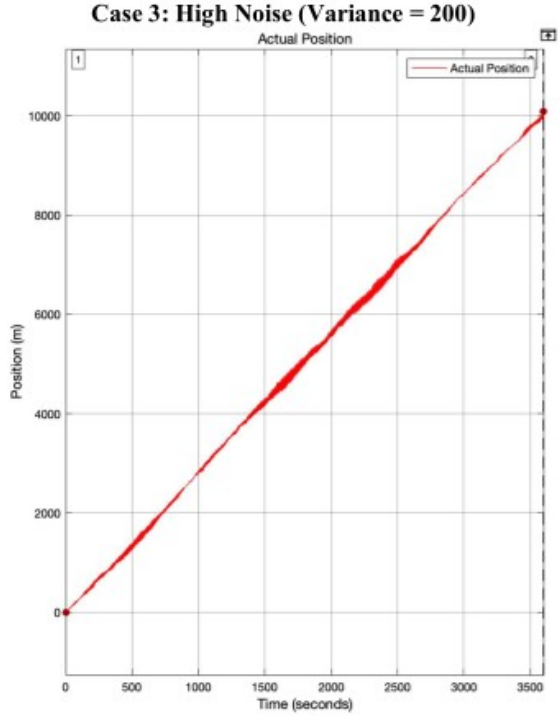

Figure 21. Model 3 $\sigma^2 = 200$ Graph

At high variance ($\sigma^2 = 200$), we observe significant impacts on system performance:

- Large amplitude oscillations in position
- More aggressive control actions to compensate for measurement errors
- Position tracking errors up to $\pm 15\sqrt{2}$ meters
- System maintains basic stability but with reduced precision
- Clear demonstration of the control system's robustness

This case represents challenging conditions, such as urban canyons or areas with poor GPS reception, where measurement reliability is significantly compromised.

Measurements

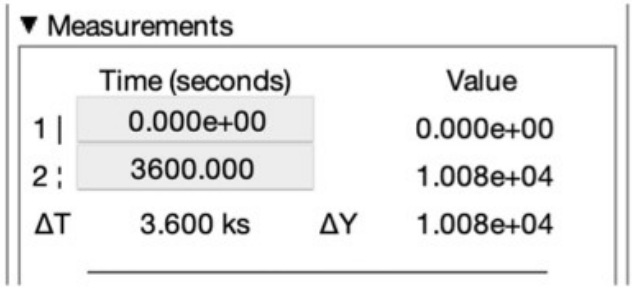

Figure 22. Model 3 $\sigma^2 = 200$ Simulation Output

Comparison with Previous Models

Model 3 represents a significant advancement over Models 1 and 2 by incorporating realistic measurement uncertainty. The key differences include:

Response Characteristics:

- Model 1: Ideal, deterministic behavior
- Model 2: Time-delayed but deterministic measurements
- Model 3: Time-delayed measurements with random errors

Control Challenges:

- Previous models dealt primarily with delay-induced instabilities
- Model 3 must also reject measurement noise while maintaining stability
- The control system must balance responsiveness against noise sensitivity

Performance Metrics:

- Position tracking accuracy now depends on noise magnitude
- Speed control becomes a trade-off between tight tracking and noise rejection
- System robustness becomes a key consideration

Conclusion

The introduction of measurement noise reveals important practical considerations for implementing such a system in real-world conditions. It demonstrates that while the basic control structure remains effective, performance expectations must be adjusted based on the **quality** of position measurements.

This analysis shows that even with significant measurement noise, our fundamental control strategy remains viable, though with degraded performance as noise increases. This robustness is crucial for practical applications where perfect measurements are never available.

V. MODEL 4 – PROPORTIONAL CONTROL, TIME DELAY, RANDOM ERRORS, INTEGRAL CONTROL

A. Model Structure

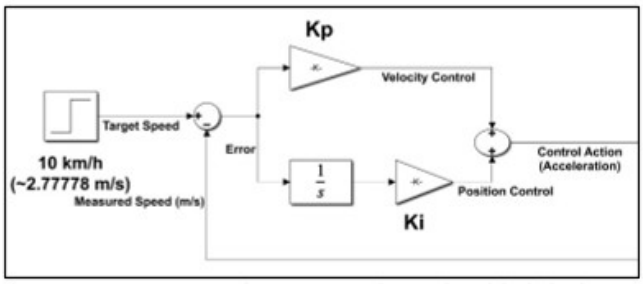

Figure 23. Integral Control Model Block Diagram

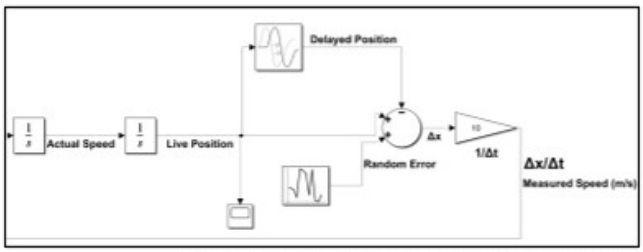

Figure 24. Integral Control Model Block Diagram Cont.

B. Motiviation

Model 4 continues to build on the previous models by introducing an integral control action. The purpose of the integral control action is to bring in another factor that the model can use to adjust the ideal speed and instructions given to the runner. In previous models, the control action was strictly proportional, meaning the model would tell the runner how to aggressively or passively adjust their speed solely based on the value of K_p, named 'Velocity Control' in Figure 23. Model 4 also uses this approach, with the additional control action K_i, named 'Position Control' in Figure 23. The position control differs from the velocity control by implementing what its name suggests, a control action based on the runner's position. As the user is running, the velocity and position control work together to provide the runner feedback based on the current speed *and* position. In previous models, the only significant factor was whether or not the runner's speed is 10 km/hr; however, with position control, the model now verifies if the runner has covered enough ground given the time elapsed. For example, 30 minutes into the race, the model will ensure that the user has run 5000 meters from the starting point, in addition to running at the target speed.

In Figure 23, the error signal is split into two paths. First looking at the lower path, the position control is multiplied by an integrator block, $\frac{1}{s}$. This integrator block converts the incoming error, which is a difference in velocity, into a position measurement, which is then scaled by K_i, the position control action value. In the upper path, the error is scaled by K_p, outputting the velocity control signal identical to the previous models. The summation of the velocity and

position control becomes the model's complete control action, which is acceleration.

Akin to past models, the control action is then integrated twice via two integrator blocks, denoted '$\frac{1}{s}$' in Figure 24. The output signal of the second integral is the actual position of the runner. Like model 3, the actual position of the runner is split into three paths, the lower signal goes to an output scope, where the position of the runner can be analyzed via a graph and the upper signal flows into a time delay block, where the runner's current position is stored for a predetermined amount of time, becoming a delayed position. This delayed position can be denoted as $x_{initial}$. The last signal, which is the runner's actual current position, flows into a summation circle, this position can be denoted as x_{final}. Intersecting at this summation circle is $x_{initial}$, x_{final}, and a random error block. The summation circle creates the expression $x_{final} - x_{initial} + Random\ error$.

As mentioned in model 3, the random error comes from a normal distribution with a variable variance. This expression is then divided by Δt, which is the predetermined time delay. The final expression becomes $\frac{x_{final}-x_{initial}+Rand \quad error}{\Delta t}$, which yields the runner's measured speed.

C. Model Analysis.

Model 4 is unique from other models in the sense that it is a third order system. Increasing the order of a system brings along pros and cons. A higher order system can reduce the steady state error; however, the system also becomes much more susceptible to instability. To isolate the effects of differing values of K_i, other values will be kept constant. K_p will be held at 0.1, the random error distribution will maintain a mean of zero and a σ^2 value of 1. Lastly, the time delay for $x_{initial}$ and Δt will both be kept at 15 seconds.

1. $\boldsymbol{K_i = 1 \times 10^{-3}}$ – Simulation Results:

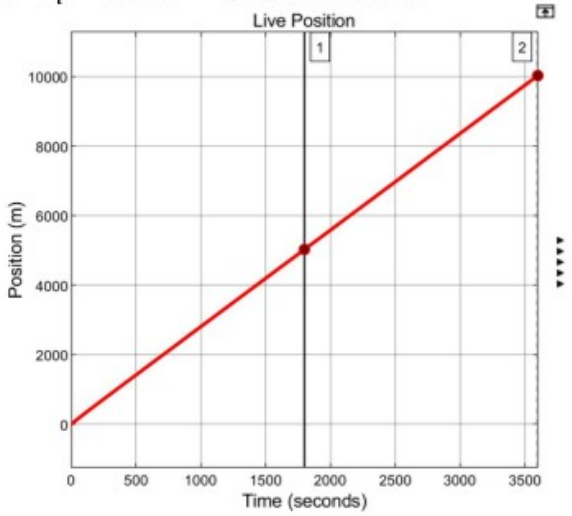

Figure 25. Model 4 K_i = 1e-3 Graph

▼Measurements

	Time (seconds)		Value
1	1800.000		5.021e+03
2	3600.000		1.002e+04
ΔT	1.800 ks	ΔY	4.999e+03

Figure 26. Model 4 K_i = 1e-3 Simulation Output

Rationale:

When $K_i = 1 \times 10^{-3}$, the system is stable, the runner can successfully complete the objective. However, this value of K_i fails to sufficiently control the position of the runner. This can be seen from the model's values at 1800 seconds. As stated previously, the runner should theoretically be 5 km away from the starting point at 1800 seconds. In Figure 26, the runner's position is approximately 5.021 km, meaning a 21-meter overshoot. Nonetheless, the model is still successful.

2. $\boldsymbol{K_i = 1 \times 10^{-4}}$ – Simulation Results:

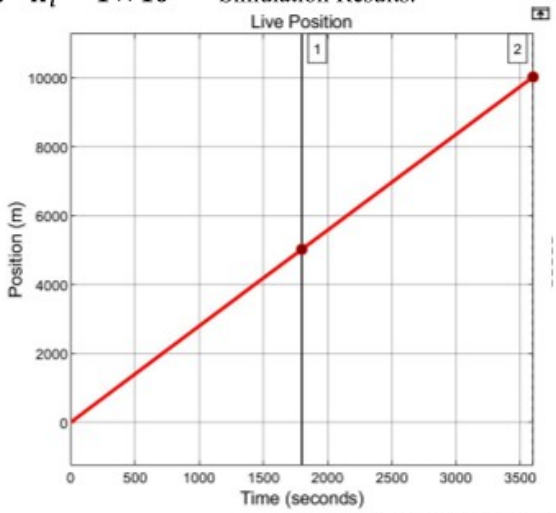

Figure 27. Model 4 K_i = 1e-4 Graph

▼Measurements

	Time (seconds)		Value
1	1800.000		5.017e+03
2	3600.000		1.002e+04
ΔT	1.800 ks	ΔY	5.003e+03

Figure 28. Model 4 K_i = 1e-4 Simulation Output

Rationale:

When $K_i = 1 \times 10^{-4}$, the system is stable, the runner can successfully complete the objective again. This value of K_i also fails to sufficiently control the position of the runner, but

to a lesser extent than the previous K_i value. This can be seen from the model's values at 1800 seconds. In Figure 28, the runner's position is approximately 5.017 km, which is an overshoot of 17 meters at the allotted time.

3. Simulation Results: $\boldsymbol{K_i = 1.5 \times 10^{-5}}$

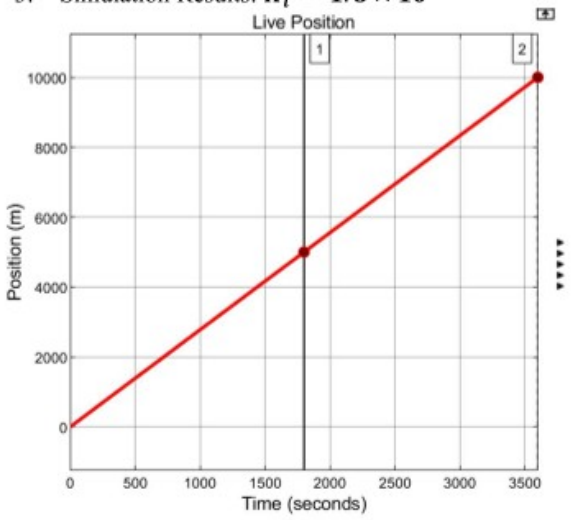

Figure 29. Model 4 K_i =1e-5 Graph

▼Measurements

	Time (seconds)		Value
1	1800.000		5.000e+03
2	3600.000		1.000e+04
ΔT	1.800 ks	ΔY	5.004e+03

Figure 30. Model 4 K_i = 1e-5 Simulation Output

Rationale:

When $K_i = 1.5 \times 10^{-5}$, the system remains stable, and the runner can complete the objective with perfect accuracy. This value of K_i successfully controls the position of the runner at the midway point in addition to the finish line. Looking at Figure 30 at 1800 seconds, the runner's position is 5 km, which is equal to the theoretical value. Also looking at the value at 3600 seconds, the runner finishes the race in the exact amount of given time.

VI. Conclusion

This surface-level analysis of control systems relating to a runner's watch attempted to analyze four differing control system models with varying components and values. Each sequential control system increased in complexity, adding to the number of variables the model must account for. Every new variable, control action, and system order contributed to making the model more sensitive to erroneous parameters, opening the door for instability. As the models increased in order, much smaller control action values were needed for the simulation to be successful.

The analysis is far from comprehensive, however. An important factor that the models do not account for is elevation. For example, a user running up a hill will naturally slow down due to the gradient. The analyzed models would not account for this and would simply tell the runner to speed up. An improved model would be able to recognize that with an uphill comes a downhill. The improved model would not tell the runner to speed up on the uphill, knowing that they will make up for lost time by naturally running faster on the downhill.

A continued study would incorporate the suggestions made above, as well as perform a more in-depth analysis into the ideal time delay value. Depending on the size of a time delay, the GPS may record the distance between two points as a tangent, when in reality the user may have been running along a curve or made a 90-degree turn.

VII. Team Members

Hocine Filali:

Hocine's contributions to the report included constructing the Simulink models used throughout the report. Hocine also contributed to the report by analyzing Model 4 and writing the conclusion. Hocine is an undergraduate student studying Systems & Industrial Engineering at George Mason University, with a concentration in mechanical design. Hocine is 21 years old and is an unfortunate supporter of Arsenal Football Club.

Amr Hamza:

Amr contributed to the report by analyzing the first two models: Model 1 and Model 2. Model 1 is the model of proportional gain, while Model 2 is the model of proportional gain with time delay. Amr is an undergraduate student studying Systems & Industrial Engineering at George Mason University, concentrating in software engineering and artificial intelligence. Moreover, Amr is 23 years old, and his hobbies are swimming, exercising, playing and watching soccer.

Adewale Adekambi:

Adewale's contributions to the report included writing the abstract, the introduction, and background. Adewale also contributed to the report by analyzing Model 3, the system accounting for GPS measurement errors. Adewale is an undergraduate student studying Systems & Industrial Engineering at George Mason University, with a concentration in electrical engineering. In addition, Wale is 22 years old and enjoys playing tennis and exercising.