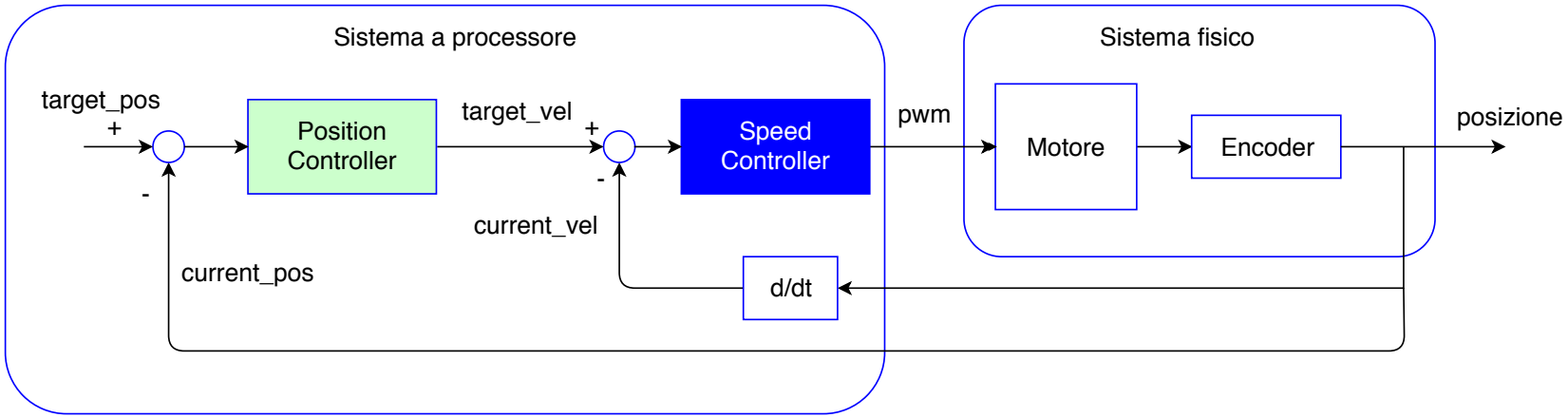
Controllo in posizione con profilo di velocità

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

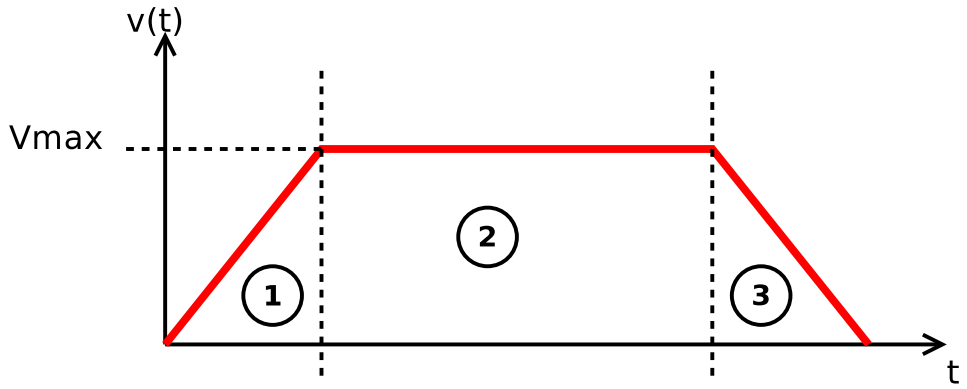
Programmazione Sistemi Robotici

Controllore di posizione



- La tecnica di controllo alternativa al semplice controllo “P” prevede l’uso di un **algoritmo software** che genera il **profilo di velocità** da seguire per poter raggiungere la posizione target.

Profilo di Velocità



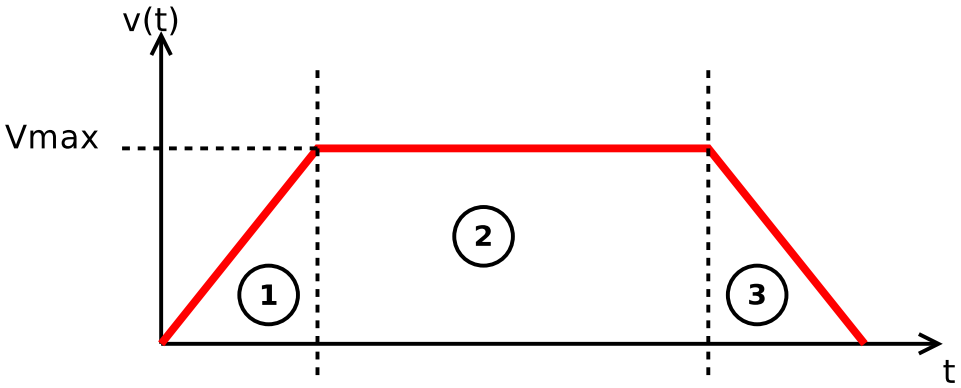
- I tipici profili di velocità dei sistemi meccanici prevedono tre fasi:
  1. Una prima fase di **accelerazione**, parametro $a_1 = cost$
  2. Una fase di *crociera* a **velocità costante**, $a_2 = 0$, $v = V_{max}$
  3. Una fase di **decelerazione**, con parametro $a_3 = cost$
- L'obiettivo è far sì che, al termine della fase di decelerazione, la massa da controllare si trovi effettivamente alla **posizione target**
- Il controllore è totalmente determinato dai **parametri** a_1, V_{max} e a_3, più il tempo di campionamento ΔT.

Profilo di Velocità e Moto uniformemente accelerato

Per determinare l'algoritmo del controllore, useremo le formule del **moto uniformemente accelerato** che qui ricordiamo:

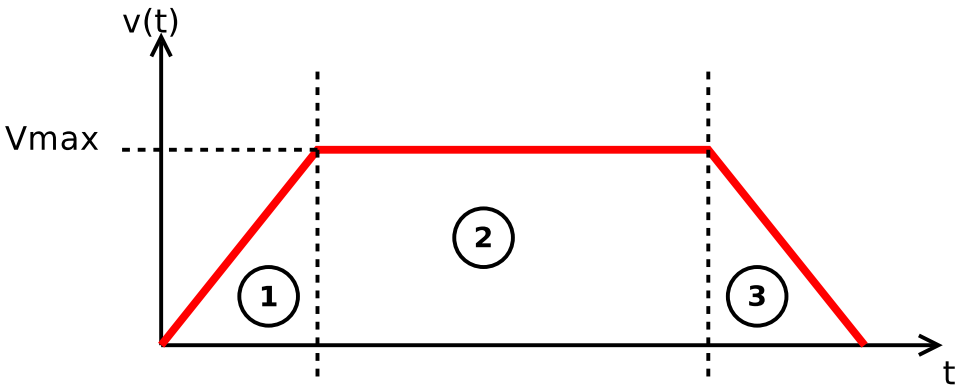
$$\begin{aligned} a &= cost \\ v(t) &= v(0) + at \\ x(t) &= x(0) + v(0)t + \frac{1}{2}at^2 \end{aligned}$$

Implementazione delle fasi



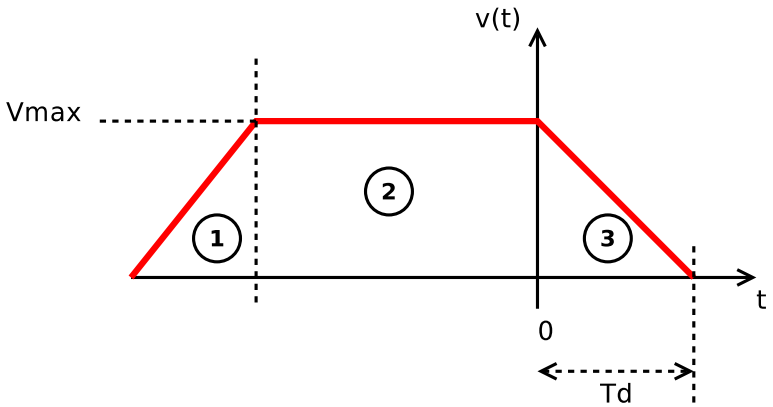
- Le fasi 1 e 2 sono abbastanza semplici: si tratta di incrementare la velocità, ad ogni step di simulazione, del valore $a_1 \Delta T$, fino alla saturazione V_{max}:

Implementazione delle fasi



- Per implemntare la fase 3 è necessario identificare il punto in cui la fase 3 stessa **deve cominciare**
- Poichè possiamo conoscere la **distanza** (errore) tra la posizione corrente $x(t)$ e la posizione target $d = X_{target} - x(t)$, possiamo usare questa informazione per calcolare **a quale distanza deve iniziare la decelerazione**
- Determiniamo quindi la **distanza di decelerazione**

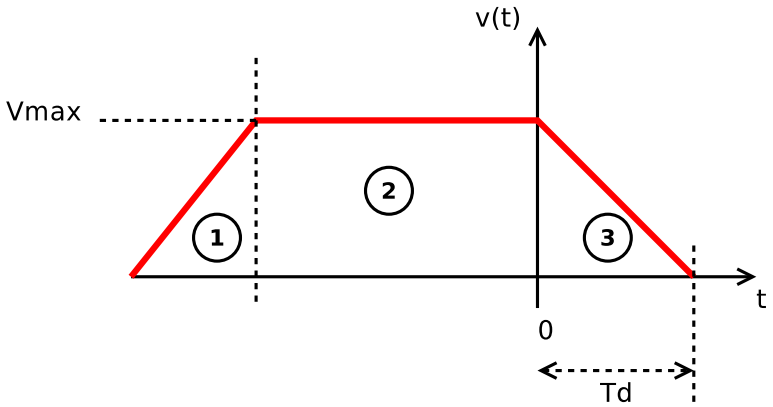
Implementazione della fase 3



- Operiamo una **traslazione temporale** e consideriamo il tempo 0 l'inizio della fase 3
- Usando la formula del moto uniformemente accelerato, calcoliamo il **tempo necessario per arrivare a velocità zero** (tempo di decelerazione) T_D:

$$\begin{aligned} v(T_D) &= v(0) + a_3 T_D \\ 0 &= V_{max} + a_3 T_D \\ T_D &= -\frac{V_{max}}{a_3} \end{aligned}$$

Implementazione della fase 3



$$
\begin{aligned}
v(T_D) &= v(0) + a_3 T_D \\
0 &= V_{max} + a_3 T_D \\
T_D &= -\frac{V_{max}}{a_3}
\end{aligned}
$$

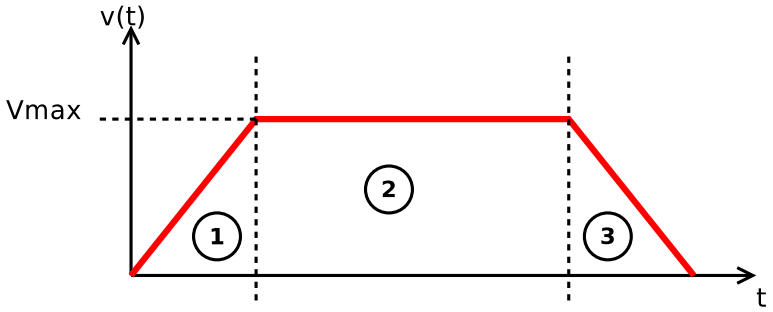
- Il segno “meno” è normale perchè, essendo a_3 **negativo** (è una decelerazione), accade che T_D risulti alla fine **positivo**.

Implementazione della fase 3

- Determiniamo a questo punto la **distanza di decelerazione** $X_D = x(T_D)$:
- Se la distanza dal target è inferiore a X_D allora siamo nella fase di decelerazione.

$$
\begin{aligned}
x(T_D) &= x(0) + v(0)T_D + \frac{1}{2}a_3 T_D^2 \\
x(T_D) &= 0 + V_{max} T_D + \frac{1}{2}a_3 T_D^2 \\
x(T_D) &= V_{max}(-\frac{V_{max}}{a_3}) + \frac{1}{2}a_3(-\frac{V_{max}}{a_3})^2 \\
x(T_D) &= -\frac{V_{max}^2}{a_3} + \frac{1}{2}a_3\frac{V_{max}^2}{a_3^2} \\
x(T_D) &= -\frac{V_{max}^2}{a_3} + \frac{1}{2}\frac{V_{max}^2}{a_3} \\
X_D &= -\frac{V_{max}^2}{2a_3}
\end{aligned}
$$

Implementazione della fase 3



- Durante la fase 3, il controllore deve fornire la velocità alla quale la massa deve andare
- Poichè conosciamo la distanza tra posizione target e posizione corrente usiamo questa informazione per determinare la velocità zero
- Dobbiamo quindi trovare una funzione del tipo

$$v(t) = f(e(t)), e(t) = X_{target} - x(t)$$

- A tale scopo usiamo ancora una volta le equazioni della cinematica applicate alla **sola fase di decelerazione**

Implementazione della fase 3

- Consideriamo il tempo 0 l'inizio della fase della decelerazione, quindi $v(0) = V_{max}$ e $x(0) = 0$, abbiamo:

$$\begin{aligned} v(t) &= V_{max} + a_3 t \\ x(t) &= V_{max} t + \frac{1}{2} a_3 t^2 \end{aligned}$$

- Poniamo, per brevità, $x = x(t)$ e $v = v(t)$ e calcoliamo il tempo dalla prima equazione:

$$t = \frac{v - V_{max}}{a_3}$$

- e sostituiamolo nella seconda:

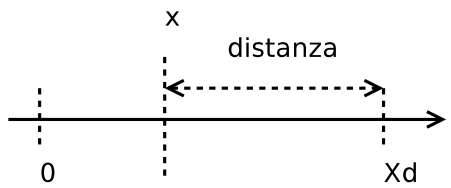
Implementazione della fase 3

$$
\begin{aligned}
x &= V_{max}\frac{v - V_{max}}{a_3} + \frac{1}{2}a_3\left(\frac{v - V_{max}}{a_3}\right)^2 \\
x &= \frac{V_{max}v - V_{max}^2}{a_3} + \frac{v^2 - 2vV_{max} + V_{max}^2}{2a_3} \\
2a_3x &= 2V_{max}v - 2V_{max}^2 + v^2 - 2V_{max}v + V_{max}^2 \\
2a_3x &= -2V_{max}^2 + v^2 + V_{max}^2 \\
2a_3x &= v^2 - V_{max}^2 \\
v &= \sqrt{V_{max}^2 + 2a_3x}
\end{aligned}
$$

- x è la distanza percorsa a partire **dall'inizio della fase di decelerazione**
- Tuttavia noi possediamo la **distanza dal target**, quindi ...

Implementazione della fase 3

$$v = \sqrt{V_{max}^2 + 2a_3 x}$$



- x è pari alla **distanza di decelerazione** X_D meno la **distanza dal target**:

$$x = X_D - (target\_pos - current\_pos)$$

- pertanto ...

$$v = \sqrt{V_{max}^2 + 2a_3(X_D - (target\_pos - current\_pos))}$$

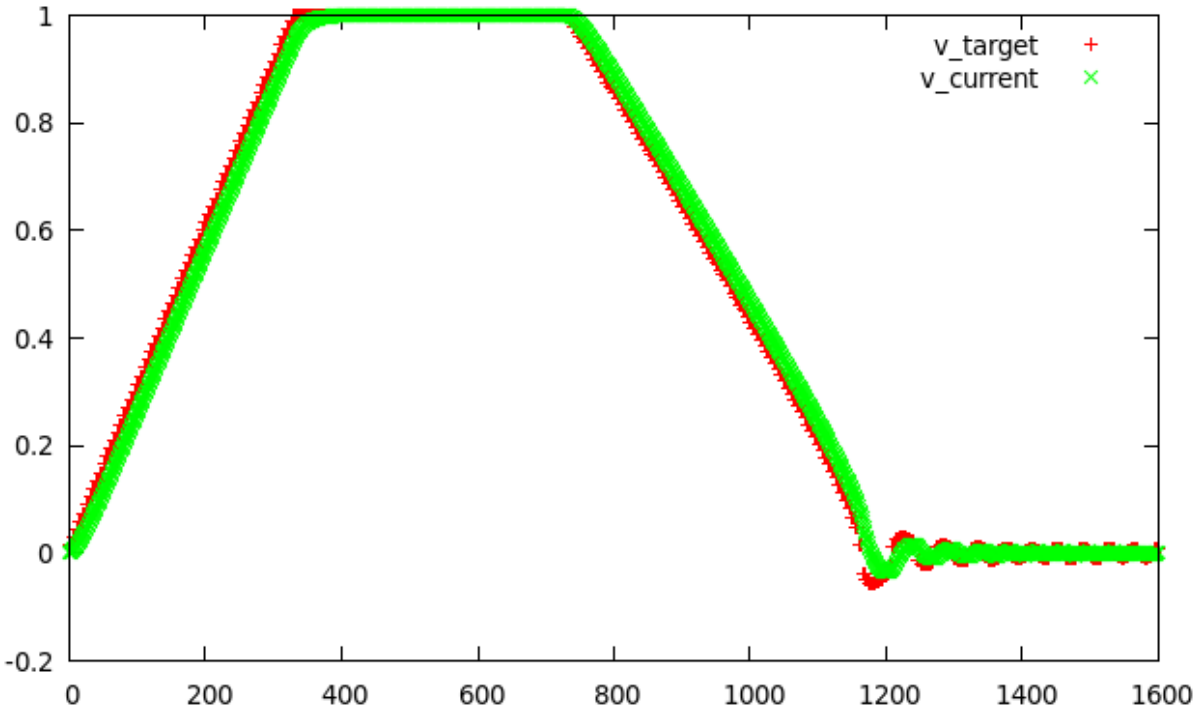
- se a_3 la si considera in valore assoluto abbiamo:

$$v = \sqrt{V_{max}^2 - 2a_3(X_D - (target\_pos - current\_pos))}$$

Andamento della velocità

L'andamento della velocità, con questo tipo di controllo, ed una distanza di $8m$, è

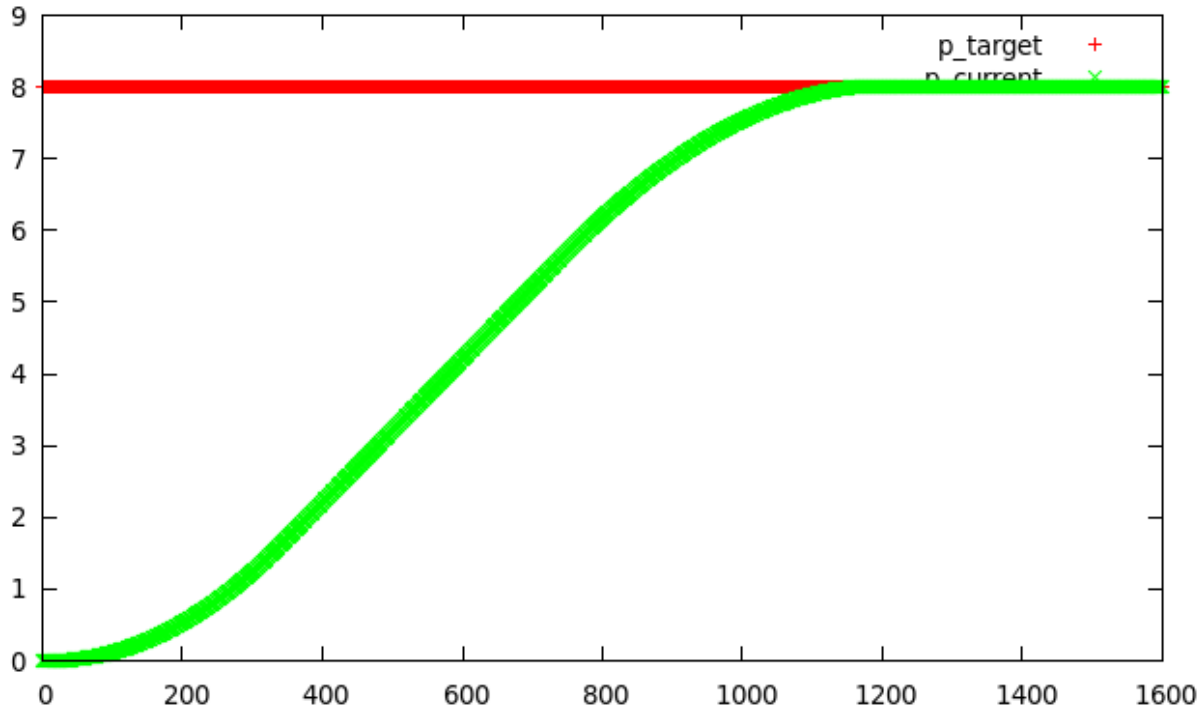
$$a_1 = 0.3m/s^2, V_{max} = 2m/s, a_3 = 0.2m/s^2$$



Andamento della distanza

L'andamento della distanza, con questo tipo di controllo, ed una distanza da percorrere di $8m$, è

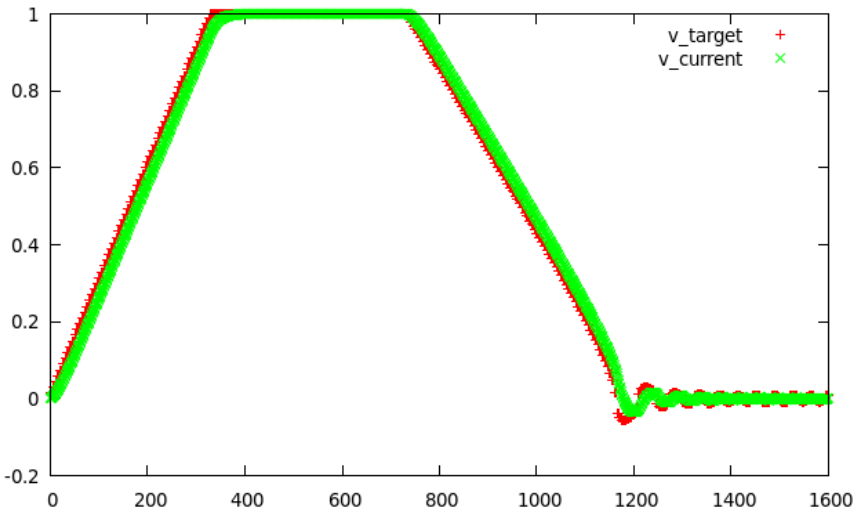
$$a_1 = 0.3m/s^2, V_{max} = 2m/s, a_3 = 0.2m/s^2$$



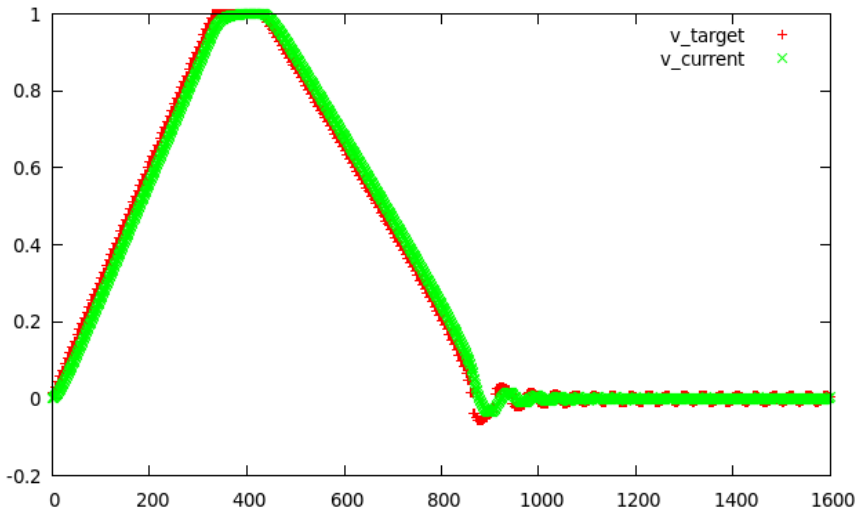
Fasi e Durata

- La durata delle fasi di accelerazione e decelerazione è funzione diretta delle costanti di accelerazione e decelerazione
- Una volta impostata a_1 e a_3, tali durate sono **sempre uguali**
- Al variare della distanza da percorrere, varierà pertanto la durata della fase a velocità costante

target $= 8m$



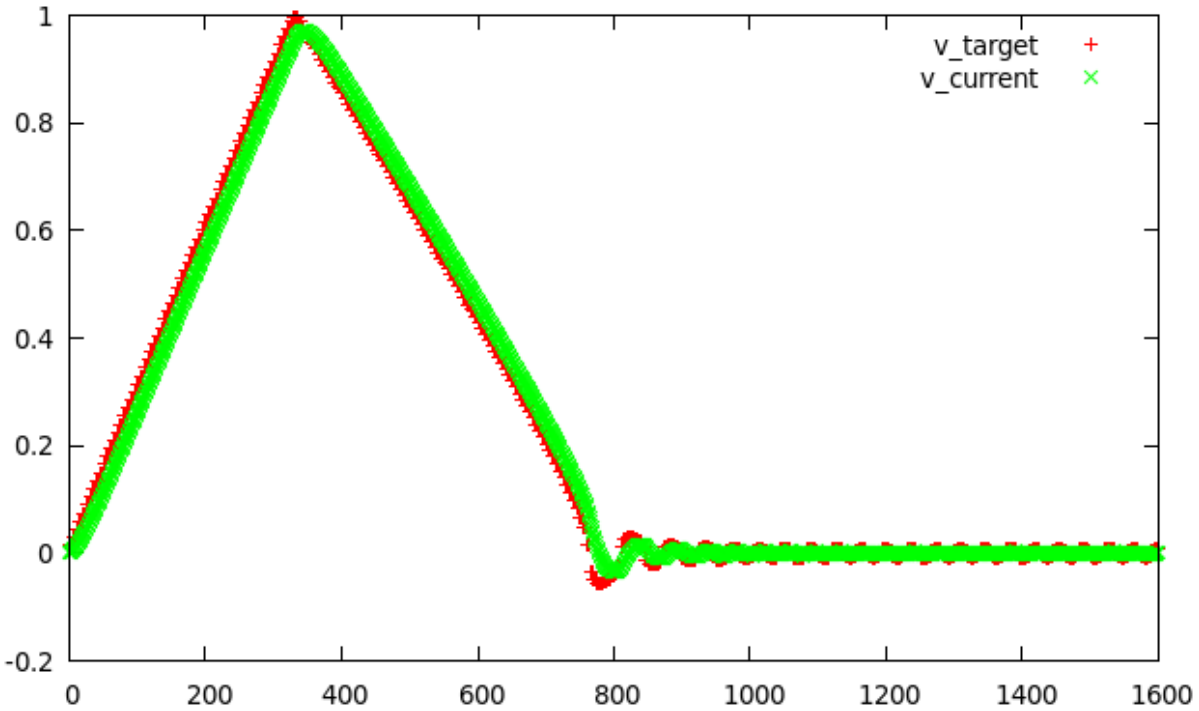
target $= 5m$



Fasi e Durata

- La fase a velocità costante potrebbe addirittura sparire nel caso la distanza sia particolarmente “corta”
- In tal caso il trapezio degenera in un **triangolo**

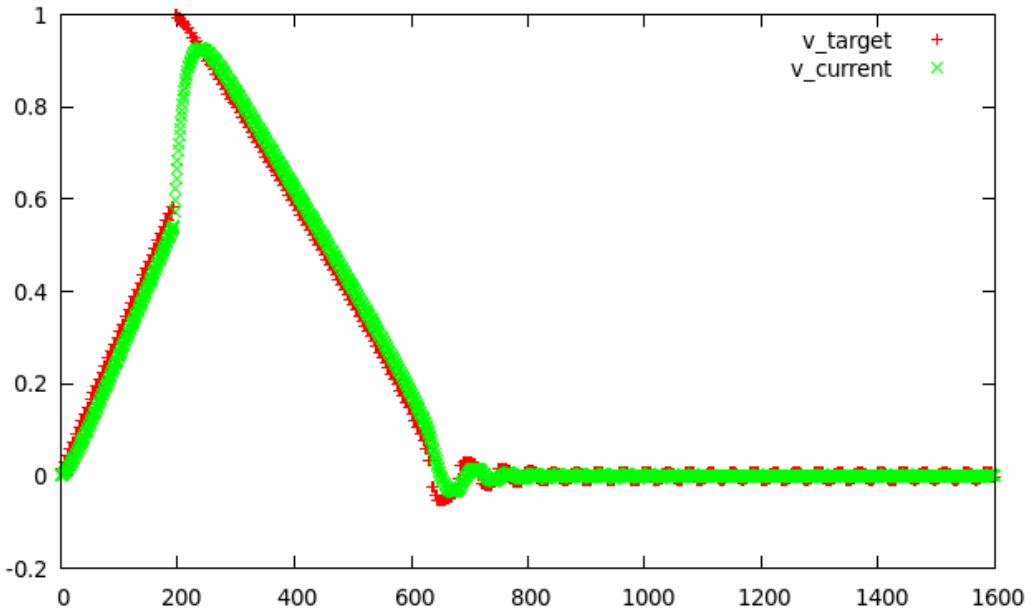
$$target = 4m$$



Degenerazione del trapezio e discontinuità

- Tuttavia se distanza è ulteriormente “corta”, le fasi 1 e 3 si *accavallano*
- In tal caso, l’algortimo descritto presenta un fastidioso punto di discontinuità e il comportamento diventa il seguente:

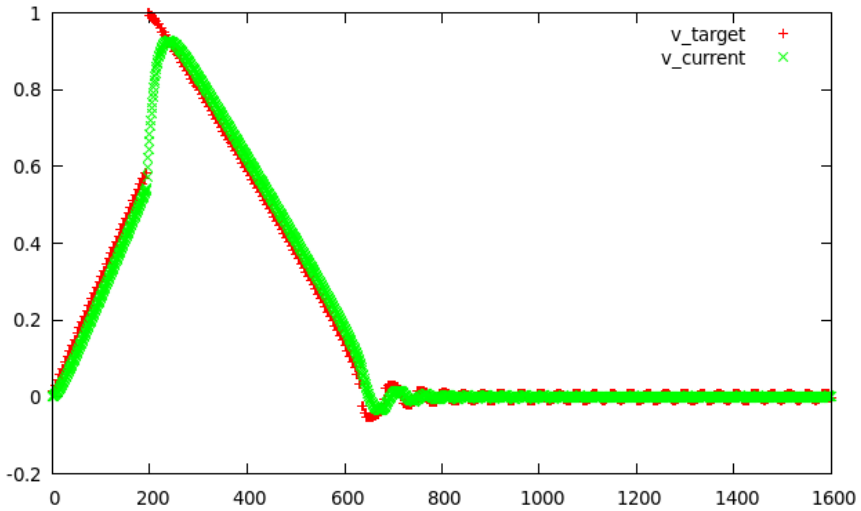
$$target = 3m$$



- Il problema è dato dal fatto che siamo già entrati nella **distanza di decelerazione** senza aver **finito la fase di accelerazione**
- La soluzione consiste nel **prolungare la fase di accelerazione** fin quando il segmento crescente non **incontra** il tratto di decelerazione

Soluzione del problema della discontinuità

$target = 3m$

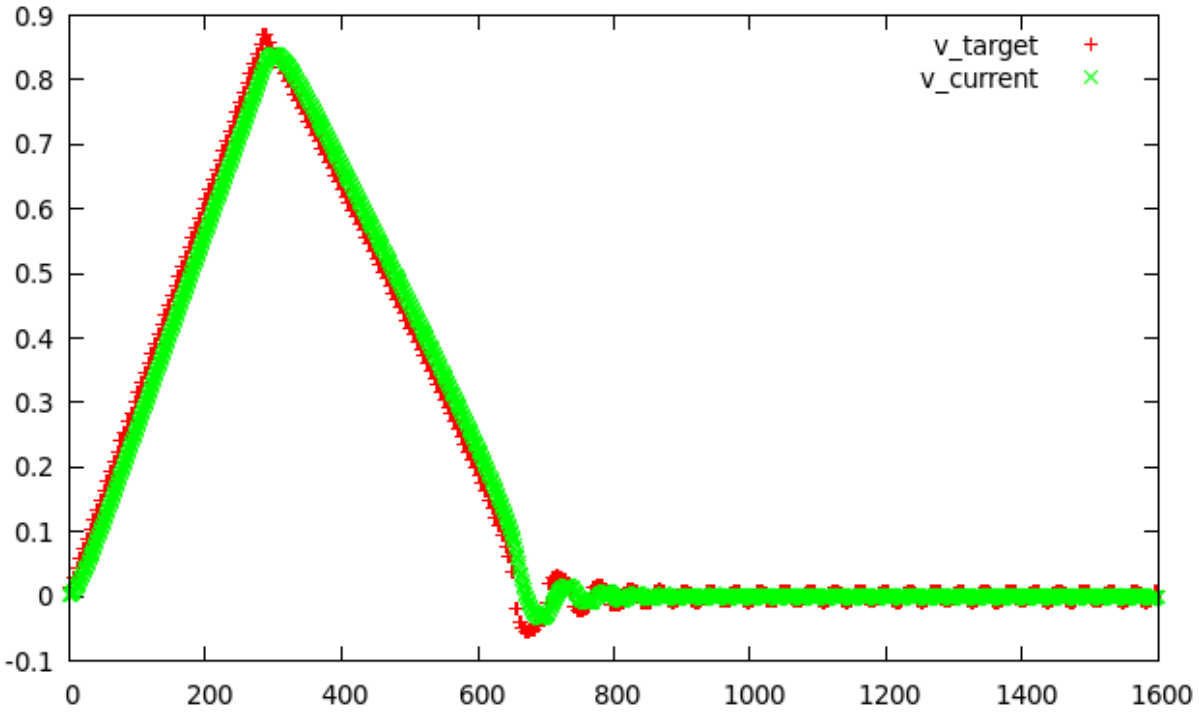


- La fase 3 deve essere caratterizzata da una **invariabilità**: **la velocità da raggiungere deve essere minore della velocità corrente**, altrimenti non ha senso parlare di *decelerazione*
- Nel caso in figura accade proprio il contrario: quando inizia la fase di decelerazione, la velocità calcolata **è maggiore** della velocità corrente
- Possiamo pertanto sfruttare questa proprietà per "far continuare" l'accelerazione fino all'incontro con il segmento di decelerazione

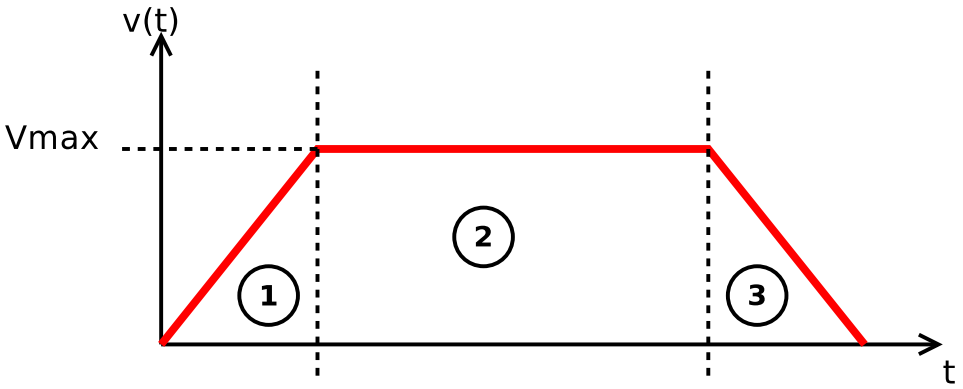
Test con distanza corta

- A questo punto, anche sulle distanze “corte”, l’algoritmo funziona correttamente:

$$target = 3m$$

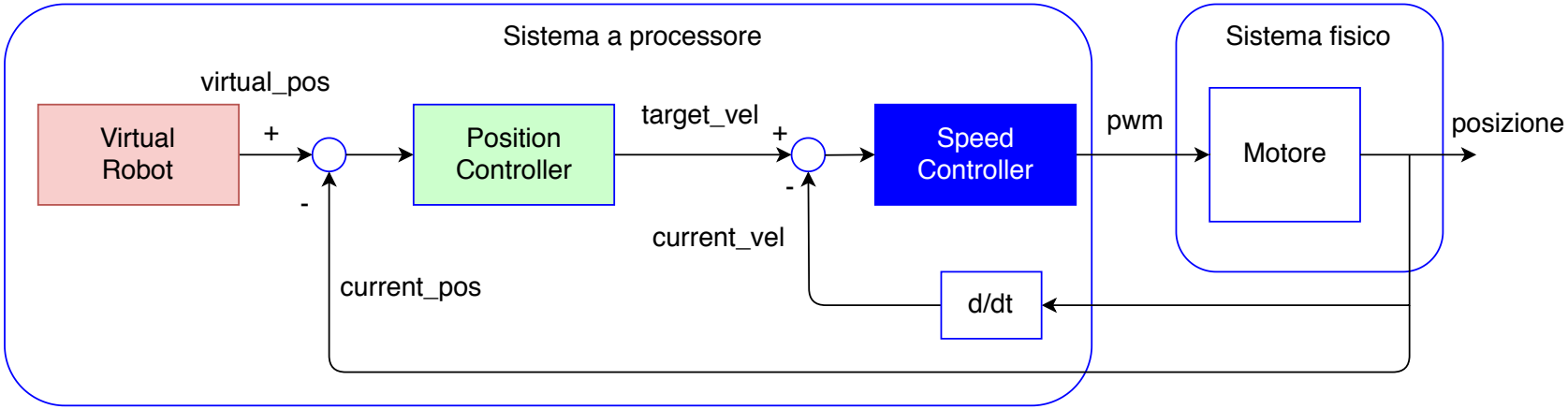


Controllo con “virtual robot”



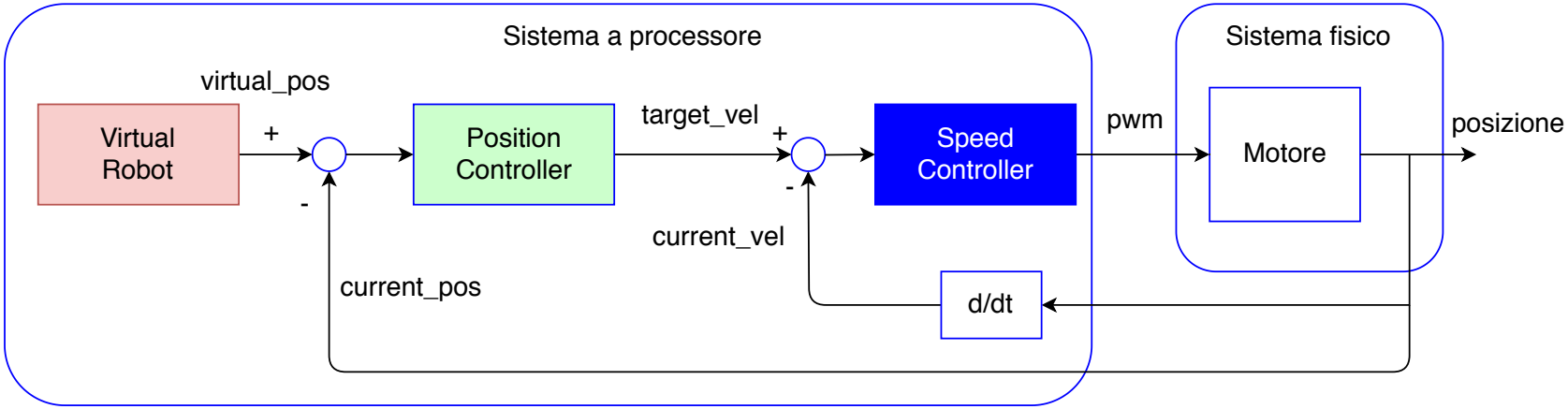
- Immaginiamo un robot “virtuale” che si muove idealmente lungo la traiettoria desiderata
- Possiamo derivare una **legge oraria**, basata sulle equazioni del moto uniformemente accelerato, che implementa il profilo trapezoidale di velocità
- Tale legge oraria fornisce la **posizione teorica** $\tilde{x}(t)$ dove dovrebbe trovarsi, all’istante t il robot

Controllo con “virtual robot”



- Usiamo il “classico” controllo: P sulla posizione e PI sulla velocità
- Un modulo che implementa il moto del *virtual robot* genera, istante per istante, la **virtual_pos**
- Essa rappresenta dove si trova il robot virtuale e dove **dovrebbe trovarsi** il robot reale
- Il robot reale cerca di raggiungere la virtual_pos che tuttavia **cambia continuamente**
- Il moto si conclude al raggiugimento della **posizione finale** del robot virtuale e (conseguentemente) del robot reale

Controllo con “virtual robot”



- Il sistema lavora mantenendo costantemente un **errore di posizione non nullo**
- Concettualmente, il robot virtuale “scappa” e il robot reale “lo insegue”
- Il robot reale è dunque “in ritardo” rispetto al robot virtuale e dunque arriverà un po’ dopo al target di posizione finale
- La costante K_P del Position Controller permette regolare tale ritardo

Cinematica “Virtual Robot”

if $TargetPos - VirtualRobotPos < DecelDistance$ **then**
    // We are in the deceleration phase
    $tempAccel \leftarrow a_3$;
end
else
    // We are in the acceleration (or constant-speed) phase
    $tempAccel \leftarrow a_1$;
end

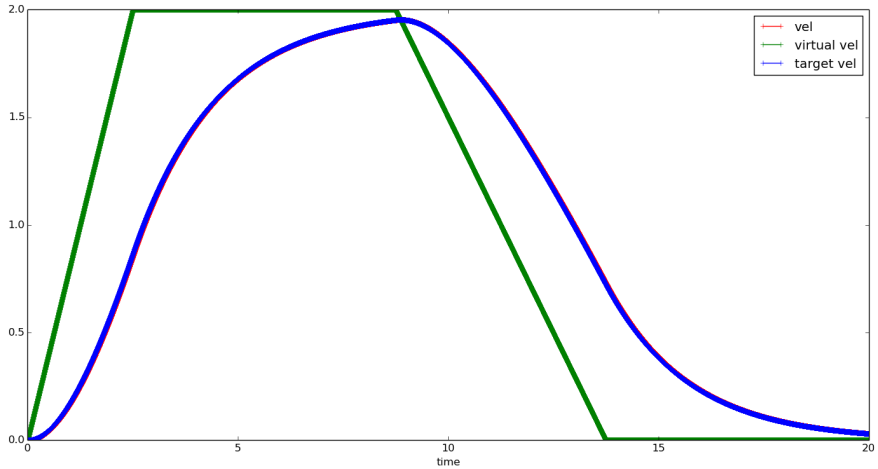
$VirtualRobotSpeed \leftarrow VirtualRobotSpeed + tempAccel \cdot \Delta T$;

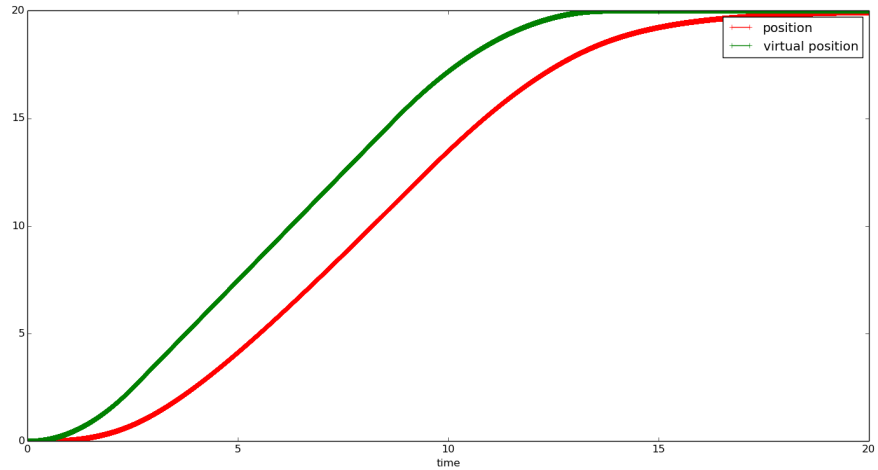
if $VirtualRobotSpeed > V_{max}$ **then**
    $VirtualRobotSpeed \leftarrow V_{max}$;
    $tempAccel \leftarrow 0$;
end
if $VirtualRobotSpeed < 0$ **then**
    $VirtualRobotSpeed \leftarrow 0$;
    $tempAccel \leftarrow 0$;
end

$VirtualRobotPos \leftarrow VirtualRobotPos + VirtualRobotSpeed \cdot \Delta T + \frac{1}{2} \cdot tempAccel \cdot \Delta T^2$;

Controllo con “virtual robot”

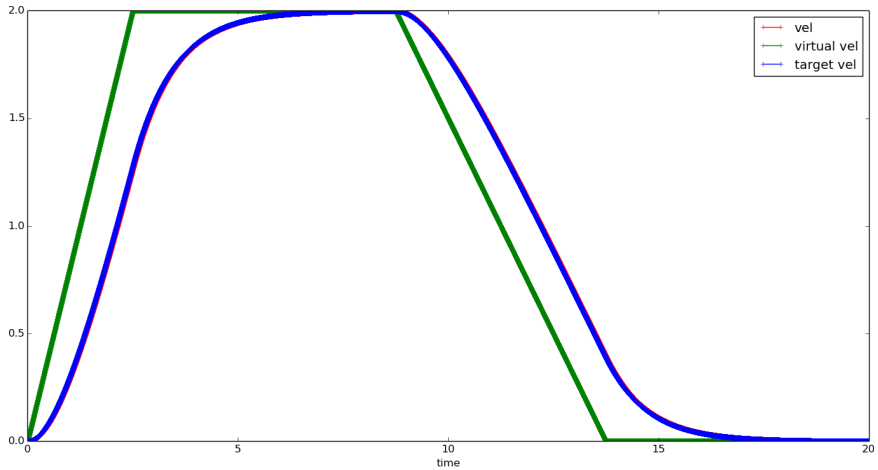
Ritardo Elevato, $KP_{POS} = 0.5$

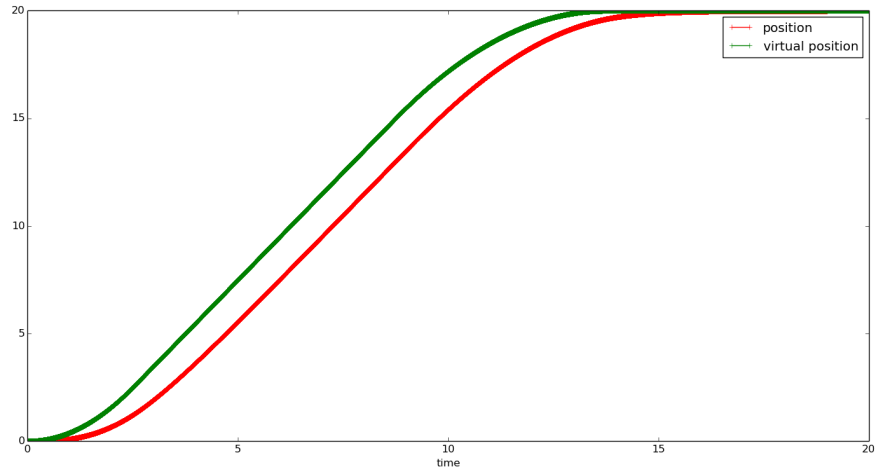




Controllo con “virtual robot”

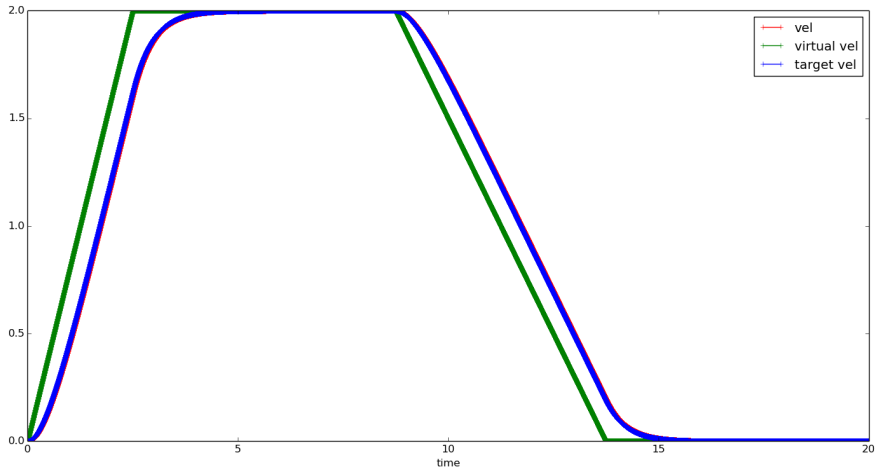
Ritardo Medio, $KP_{POS} = 1$

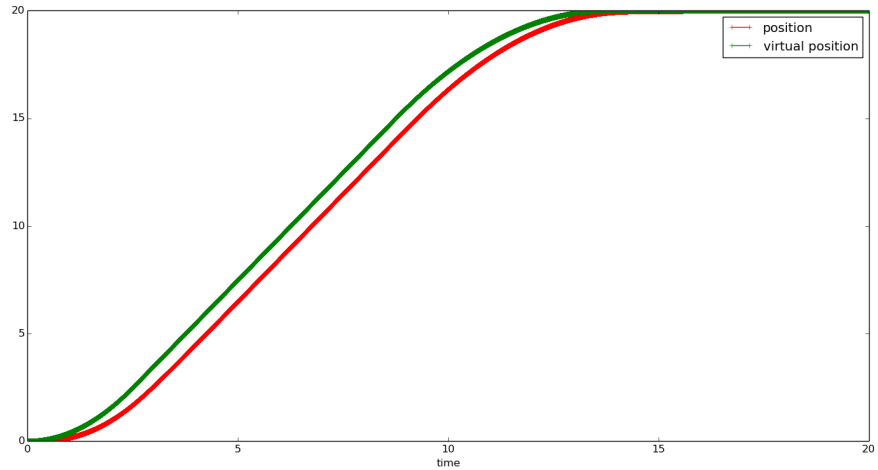




Controllo con “virtual robot”

Ritardo Basso, $KP_{POS} = 2$





Controllo in posizione con profilo di velocità

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

Programmazione Sistemi Robotici