Intelligent distributed systems

Daniele Fontanelli

Department of Industrial Engineering
University of Trento
E-mail address: *daniele.fontanelli@unitn.it*

2022/2023

UNIVERSITÀ DI TRENTO | Dipartimento di Ingegneria Industriale

Outline

Outline

Discretisation of Systems

In modern control systems a *digital discrete-time* system is interfaced to a *continuous-time* physical plant.
In a *distributed control system*, the interface takes place for different reasons: a) to share information in digital networks; b) to control plants using digitally implemented controllers; c) to control networked systems.
To show the problems related to digitalisation of systems, we will make use of the classic *feedback control* and then we will extend the results on the general case.

Discretisation of Systems

Components

Usually *three* actors take place in a discretisation process:

- *Hold*, which is used to convert a discrete-time sequence $\{u_k\}$ into a continuous-time sequence suitable for the physical plant: $u(t) = u_k$, for $k\Delta_t \leq t \leq (k+1)\Delta_t$, where Δ_t is the sampling time;
- *Physical system*, usually described by a set of linear or nonlinear differential equations;
- *Sampler*, that is the device that creates the discrete-time sequence.

Sometimes an *Anti-aliasing filter*, which prepares the continuous-time output signal, is adopted. This is certainly true for *high-sampling rates*.

Discretisation of Systems

Components

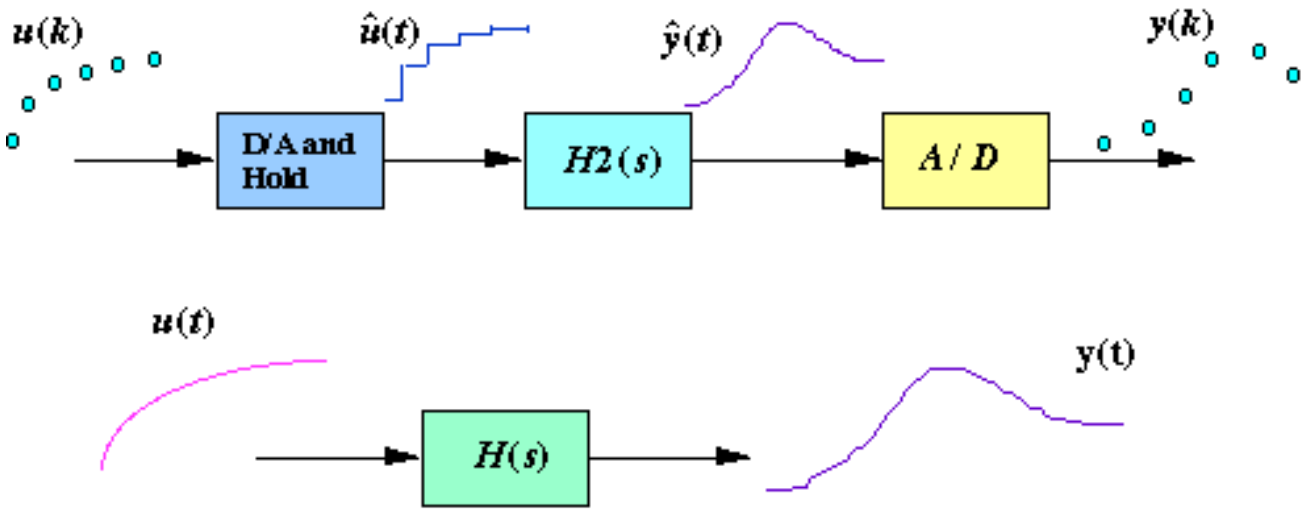


Figure: Components of a digital system as approximation of a continuous time behavior.

Discretisation of Systems

Components

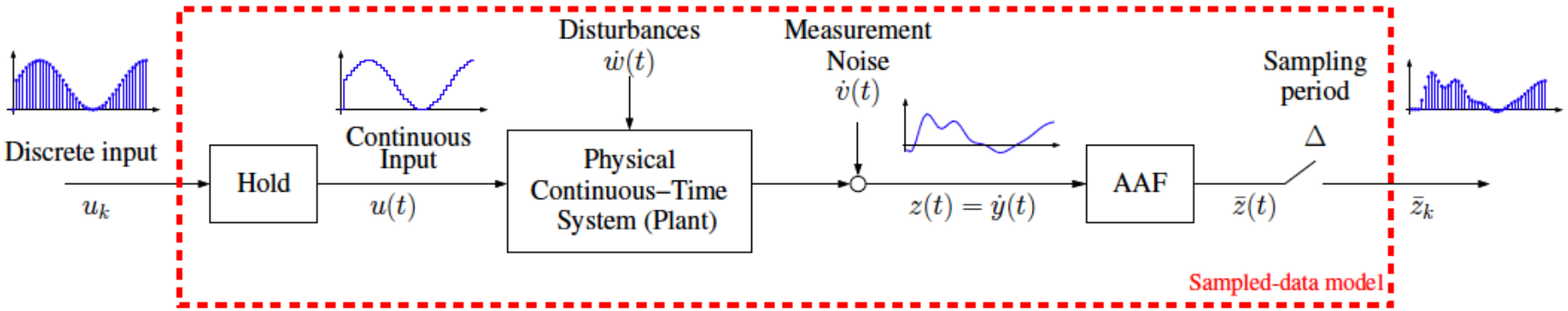


Figure: Components of a digital system with anti-aliasing filter and noise (courtesy of Goodwin, Graham C and Yuz, Juan I and Agüero, JC and Cea, Mauricio, "Sampling and sampled-data models", American Control Conference (ACC), 2010.

Outline

Discretisation of Linear Systems

Linear continuous time controllers need to be *discretised* in order to be implemented in a digital system, i.e., by an algorithm.
Discretisation is the process of transferring continuous models into *discrete* models.
In the case of a dynamic system, the models are transferred from continuous time to *discrete time*, hence obtaining a *discrete time controller*.
Therefore, the system is sampled using a well defined *sampling time* Δ_t. In digital systems, the sampling time is usually lower bounded by feasibility reasons.

Discretisation of Linear Systems

- There exist different solutions to discretise a system:
  (1) A trivial solution may be to use a sampling time that is the smaller possible, while keeping the controller continuous and using numeric tools for differential equations.
  (2) Alternatively, there are different solutions that tries to minimize the *discretisation approximations*, which can be applied to transfer functions $G(s)$ or to *state space* models.
- In the latter case, the approximation introduced are strictly related to the discrete time approximation of the integral of a continuous function.

Numerical Integration

- Consider the following transfer function and its time representation

$$\frac{Y(s)}{U(s)} = G(s) = \frac{\alpha}{s+\alpha} \Rightarrow \dot{y}(t) + \alpha y(t) = \alpha u(t) \Rightarrow \dot{y}(t) = f(y(t), u(t)).$$

- Therefore, considering the continuous–time integral, one gets

$$y(t) = \int_0^t -\alpha y(\tau) + \alpha u(\tau) d\tau,$$

that, assuming a sampling time Δ_t for the discrete–time approximation, turns to

$$y(k\Delta_t) = \int_0^{k\Delta_t - \Delta_t} -\alpha y(\tau) + \alpha u(\tau) d\tau + \int_{k\Delta_t - \Delta_t}^{k\Delta_t} -\alpha y(\tau) + \alpha u(\tau) d\tau = y(k\Delta_t - \Delta_t) + A,$$

where

$$A = \int_{k\Delta_t - \Delta_t}^{k\Delta_t} -\alpha y(\tau) + \alpha u(\tau) d\tau$$

is the area under the function $-\alpha y(\tau) + \alpha u(\tau)$ between $k\Delta_t - \Delta_t$ and $k\Delta_t$.

Shift Operator

- As previously depicted, a linear *differential equation* expressed in continuous time is expressed in discrete time by means of a linear *difference equation*, where the difference is supposed with respect to time.
- An analogous of the *differential–operator* of continuous time equations can be defined for linear difference equations with constant coefficients.
- The *forward–shift operator* is denoted by q, i.e.,

$$qf(k) = f(k+1),$$

where the sampling period is assumed to be the *time unit*, i.e.,
if $f(t)$ is sampled every Δ_t seconds and $f(k) \leftrightarrow f(k\Delta_t)$, then

$$qf(k) = f(k+1) \leftrightarrow f(k\Delta_t + \Delta_t).$$

- The *backward–shift operator*, or *delay operator*, is denoted by q^{-1}, i.e.,

$$q^{-1}f(k) = f(k-1).$$

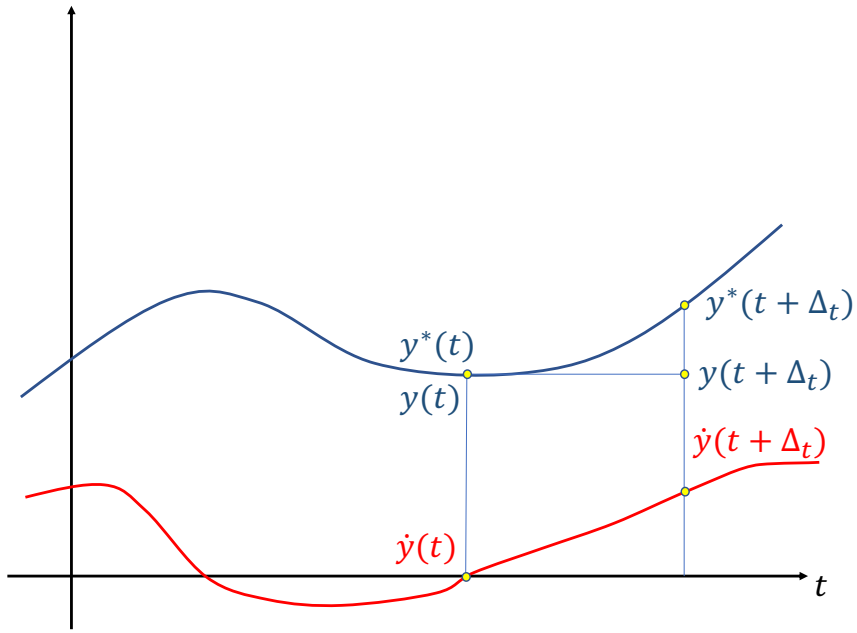
Discretisation

Euler Integration Rule

The first approximation of the integral is given by the *Euler's method*

$$\dot{y}(t) = f(y(t), u(t)) = \frac{dy(t)}{dt} \approx \frac{y(t + \Delta_t) - y(t)}{\Delta_t},$$

also known as *forward difference* or *forward rectangular rule*.

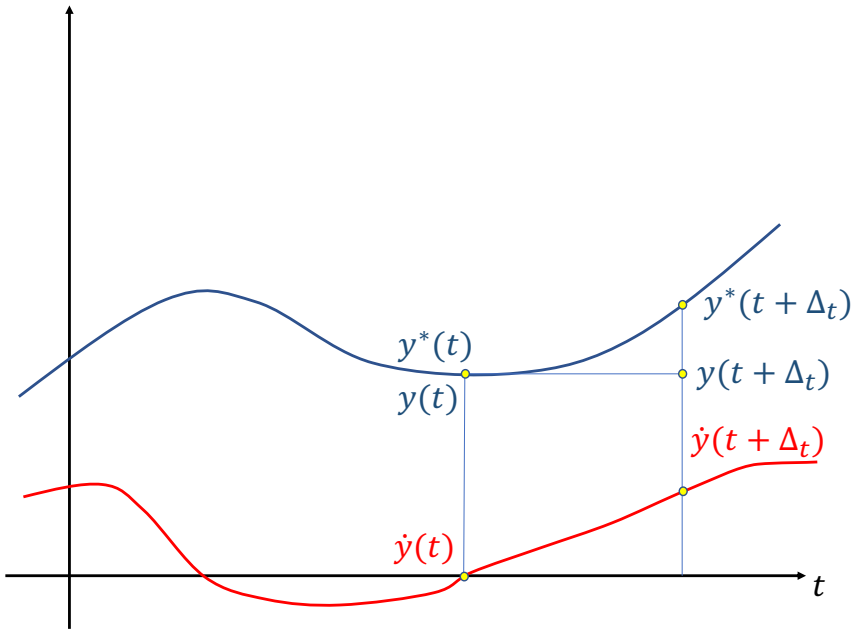


Discretisation

Euler Integration Rule

Hence

$$y(t+\Delta_t) = y(t) + \Delta_t \dot{y}(t) = y(t) + \Delta_t f(y(t), u(t)).$$



Discretisation

Euler Integration Rule

- An alternative interpretation of the forward difference method comes with the shift operator:

$$\frac{dy(t)}{dt} \approx \frac{y(t+\Delta_t) - y(t)}{\Delta_t} = \frac{q-1}{\Delta_t} y(t),$$

or, by using the discrete index k,

$$\frac{dy(k)}{dt} \approx \frac{q-1}{\Delta_t} y(k).$$

Discretisation

Euler Integration Rule

- Therefore, by recalling the fact that s corresponds to the differential operator, one gets

$$\frac{dy(k)}{dt} \approx \frac{q-1}{\Delta_t} y(k) \Rightarrow s \leftarrow \frac{q-1}{\Delta_t},$$

that is, *it is sufficient to substitute s in $G(s)$ with the proper shift operator equation*.

Remark

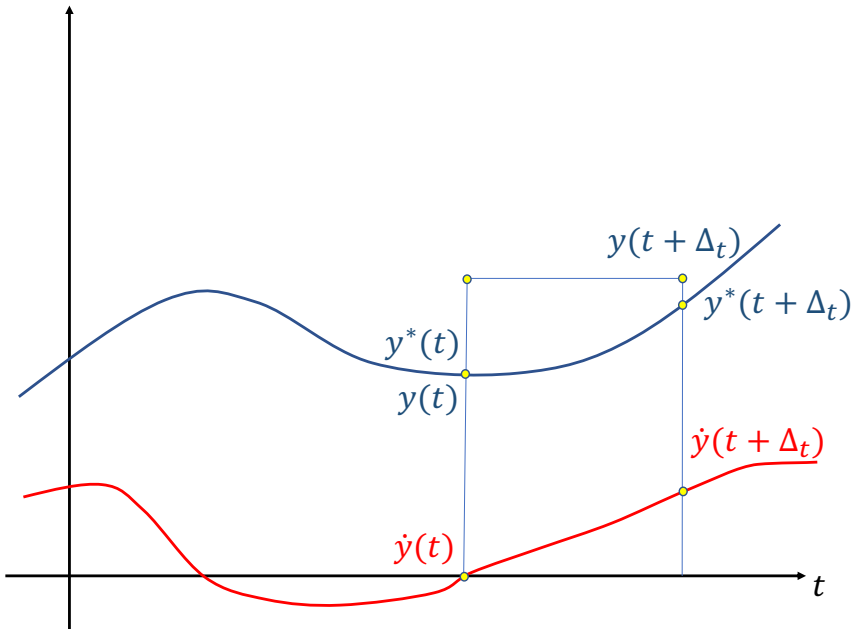
Notice that the δ-operator, i.e., $\delta = \frac{q-1}{\Delta_t}$, is not given only by the Euler integration rule. The δ-operator simply re-parametrizes discrete models through a transformation of general validity.

Discretisation

Backward Rectangular Rule

The second method, the inverse of the previous, is given by the *backward difference* or *backward rectangular rule*

$$\frac{dy(t)}{dt} \approx \frac{y(t) - y(t - \Delta_t)}{\Delta_t} \quad \text{or} \quad \frac{dy(t + \Delta_t)}{dt} \approx \frac{y(t + \Delta_t) - y(t)}{\Delta_t}.$$

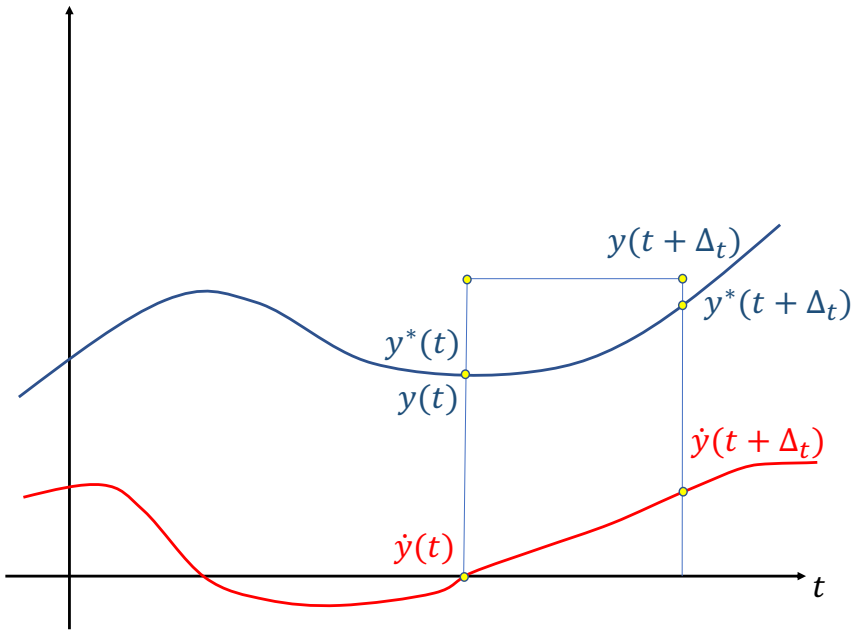


Discretisation

Backward Rectangular Rule

Hence

$$y(t+\Delta_t) = y(t) + \Delta_t \dot{y}(t+\Delta_t) = y(t) + \Delta_t f(y(t+\Delta_t), u(t+\Delta_t)).$$



Discretisation

Backward Rectangular Rule

- An alternative interpretation of the backward difference method using the shift operator is the following:

$$\frac{dy(t+\Delta_t)}{dt} \approx \frac{y(t+\Delta_t)-y(t)}{\Delta_t} = \frac{q-1}{\Delta_t} y(t),$$

or, by using the discrete index k,

$$\frac{dy(k+1)}{dt} = \frac{dqy(k)}{dt} \approx \frac{q-1}{\Delta_t} y(k).$$

Discretisation

Backward Rectangular Rule

- Therefore, by recalling the fact that s corresponds to the differential operator, one gets

$$\frac{dqy(k)}{dt} \approx \frac{q-1}{\Delta_t} y(t) \Rightarrow sq \leftarrow \frac{q-1}{\Delta_t},$$

and, more easily,

$$s \leftarrow \frac{q-1}{q\Delta_t},$$

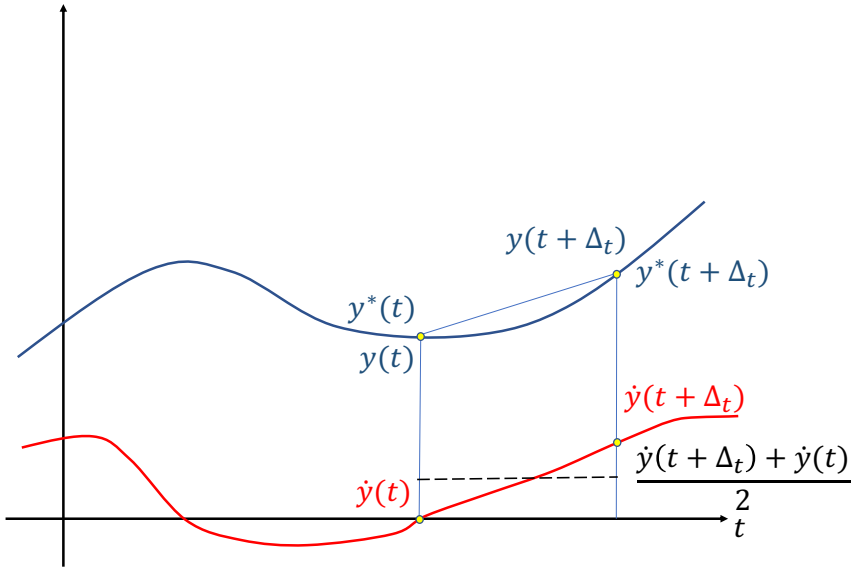
that is again sufficient to substitute s in $G(s)$ with the proper shift operator equation.

Discretisation

Trapezoidal Rule

- The third and more accurate approximation of the integral is given by the *trapezoidal rule*

$$\int_{t_1}^{t} \dot{y}(\tau)d\tau = \int_{t_1}^{t} \frac{\dot{y}(t_2)+\dot{y}(t_1)}{2}d\tau = (t-t_1)\frac{\dot{y}(t_2)+\dot{y}(t_1)}{2}, \forall t \in [t_1, t_2].$$



Discretisation

Trapezoidal Rule

- The trapezoidal rule can be interpreted as the integral of the *time derivatives*

$$\frac{1}{2}\left(\frac{dy(t+\Delta_t)}{dt}+\frac{dy(t)}{dt}\right)\approx\frac{q-1}{\Delta_t}y(t),$$

or, using the discrete index k,

$$\frac{1}{2}\left(\frac{dqy(k)}{dt}+\frac{dy(k)}{dt}\right)\approx\frac{q-1}{\Delta_t}y(k).$$

Discretisation

Trapezoidal Rule

- Therefore, by recalling the fact that s corresponds to the differential operator, one gets

$$\frac{1}{2}\left(\frac{dqy(k)}{dt}+\frac{dy(k)}{dt}\right)\approx\frac{q-1}{\Delta_t}y(k)\Rightarrow\frac{qs+s}{2}\leftarrow\frac{q-1}{\Delta_t}$$

  and, more easily,

$$s\leftarrow\frac{2}{\Delta_t}\frac{q-1}{q+1}.$$

Example of Discretisation

Euler Integration Rule

- Consider the following transfer function and its time representation

$$\frac{Y(s)}{U(s)} = G(s) = \frac{\alpha}{s+\alpha} \Rightarrow \dot{y}(t) + \alpha y(t) = \alpha u(t).$$

- Using the forward rectangular rule, one gets

$$\frac{q-1}{\Delta_t} y(k) + \alpha y(k) = \alpha u(k) \Rightarrow y(k+1) = (1 - \alpha\Delta_t)\, y(k) + \alpha\Delta_t u(k),$$

  that is the difference equation that expresses the discrete–time approximation of $Y(s) = G(s)U(s)$. Notice that the initial condition for $y(k)$ is needed.
- The same result can be obtained by simply write the equation in s and than substituting $Y(s)$ with $y(k)$, $U(s)$ with $u(k)$ and s with $\frac{q-1}{\Delta_t}$. This is very useful as soon as the transfer function has terms of the type s^n.

Example of Discretisation

Backward Rectangular Rule

- Consider the same transfer function

$$\frac{Y(s)}{U(s)} = G(s) = \frac{\alpha}{s+\alpha} \Rightarrow \dot{y}(t) + \alpha y(t) = \alpha u(t).$$

- Using the backward rectangular rule, one gets

$$\frac{q-1}{q\Delta_t} y(k) + \alpha y(k) = \alpha u(k) \Rightarrow y(k+1) = \frac{y(k)}{1+\alpha\Delta_t} + \frac{\alpha\Delta_t u(k+1)}{1+\alpha\Delta_t}.$$

  Notice that the initial condition for $y(k)$ and for $u(k)$ is needed.

- The same result can be obtained by simply write the equation in s and than substituting $Y(s)$ with $y(k)$, $U(s)$ with $u(k)$ and s with $\frac{q-1}{q\Delta_t}$. This is very useful as soon as the transfer function has terms of the type s^n.

Example of Discretisation

Trapezoidal Rule

- Consider the same transfer function

$$\frac{Y(s)}{U(s)} = G(s) = \frac{\alpha}{s+\alpha} \Rightarrow \dot{y}(t) + \alpha y(t) = \alpha u(t).$$

- Using the trapezoidal rule, one gets

$$\frac{2}{\Delta_t}\frac{q-1}{q+1}y(k) + \alpha y(k) = \alpha u(k) \Rightarrow y(k+1) = \frac{2-\alpha\Delta_t}{2+\alpha\Delta_t}y(k) + \frac{\alpha\Delta_t}{2+\alpha\Delta_t}u(k) + \frac{\alpha\Delta_t}{2+\alpha\Delta_t}u(k+1).$$

  Notice that the initial condition for $y(k)$ and for $u(k)$ is needed.

- The same result can be obtained by simply write the equation in s and than substituting $Y(s)$ with $y(k)$, $U(s)$ with $u(k)$ and s with $\frac{2}{\Delta_t}\frac{q-1}{q+1}$. This is very useful as soon as the transfer function has terms of the type s^n.

Discretisation

Hint for SISO systems

A quite simple way to automatically discretise the system is to:

(1) Substitute to the variable s in $G(s)$ the function of the variable q;

(2) Simplify the expressions of the numerator and the denominator;

(3) The denominator will be multiplied by $y(k)$, the numerator by $u(k)$;

(4) The coefficients of the two polynomials in q are the coefficients of $y(k)$, i.e., $\left(a_n q^n + a_{n-1} q^{n-1} + \cdots + a_1 q + a_0\right) y(k)$ turns to $a_n y(k+n) + a_{n-1} y(k+n-1) + \cdots + a_1 y(k+1) + a_0 y(k)$. The same for $u(k)$.

Discretisation

Remarks for SISO systems

- Usually, the shift operator is a convenient way to express the complex variable z, the variable of the *$\mathcal{Z}$–transform*, that is the discrete time counterpart of the Laplace transform.
- As soon as the $\mathcal{Z}$–transform is considered, the approximation of the trapezoidal rule is also called *Tustin's approximation* or *bilinear transformation*.
- The Tustin approximation is derived by the approximation of the continuous time delay of the sampling time Δ_t, i.e.,

$$z = e^{s\Delta_t} \approx \frac{1 + s\Delta_t/2}{1 - s\Delta_t/2},$$

which is the *Padé approximate* of the first order.

Discretisation

Remarks for SISO systems

- The most accurate discretisation algorithm among the previous is the one given by the trapezoidal rule. Indeed, it gives the best approximation of the integral operator.
- More precisely:
  - The forward difference may generate an unstable discrete–time system starting from a stable continuous–time system;
  - The backward difference may generate stable discrete–time systems starting from unstable continuous–time systems;
  - The trapezoidal rule maps continuous–time stable systems into discrete–time stable systems and unstable into unstable systems.

Outline

State space representation

Consider a time-invariant continuous-time linear system:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$z(t) = Cx(t)$$

The model can be represented in incremental form by

$$dx(t) = Ax(t)dt + Bu(t)dt$$
$$z(t)dt = Cx(t)dt$$

State space representation

Forced and unforced response

The idea is to express the evolution of the system in the *inter sampling*, i.e., during the time Δ_t.
To this end, due to the linearity of system, we can express the evolution of the system using the *unforced response*, i.e., when the input is zero and the system evolves from a generic initial state $x(0)$, and the *forced response*, i.e., when the initial condition is $x(0) = 0$ and the input is generic.
To derive the associated solutions, we will examine the results of the scalar difference equation

$$\dot{x}(t) = ax(t) + bu(t)$$

and then extend such results to the multidimensional case.

State space representation

Forced and unforced response

The *homogeneous* solution, i.e., the *unforced response*, of the differential scalar equation is simply given by

$$x_u(t) = e^{at}x(0),$$

where using the Taylor expansion around $t_0 = 0$, can be represented as

$$x_u(t) = \sum_{i=0}^{+\infty} \frac{a^i t^i}{i!} x(0).$$

Similarly, we can assume that the *unforced response* for the multidimensional case is given by

$$x_u(t) = \left(I + At + \frac{A^2 t^2}{2} + \dots \right) x(0) = \sum_{i=0}^{+\infty} \frac{A^i t^i}{i!} x(0).$$

State space representation

Forced and unforced response

Now, we have to show that such a solution is effectively a solution of the differential equation $\dot{x}_u(t) = Ax_u(t)$.
It is easy to see that

$$\dot{x}_u(t) = \left(0 + A + At + \frac{A^2t^2}{2} + \dots\right) x(0) =$$
$$= A\left(I + At + \frac{A^2t^2}{2} + \dots\right) x(0) = Ax_u(t),$$

as desired.

State space representation

Forced and unforced response

Due to the similarity here defined, we can state that

$$e^{At} = \sum_{i=0}^{+\infty} \frac{A^i t^i}{i!},$$

and then

$$x_u(t) = e^{At} x(0) = \Phi(t) x(0),$$

where $\Phi(t)$ is in general called the *state transition matrix*.

State space representation

Exercise

As an exercise, compute the *unforced response* of

$$\dot{x}_1 = x_1$$
$$\dot{x}_2 = -2x_1 - x_2 + u$$

when the initial conditions are $x_1(0) = 1$ and $x_2(0) = 2$.

State space representation

Forced and unforced response

It is now necessary to compute the *particular* solution of the differential equation, i.e., *forced response*, to derive the complete output of the system, which will be given by the *superposition principle*, i.e., the sum of forced and unforced responses.
There are several ways to compute this. One possibility is to compute *directly* the sum of the forced and unforced responses.
Consider again the scalar system

$$\dot{x}(t) = ax(t) + bu(t) \Rightarrow \dot{x}(t) - ax(t) = bu(t).$$

State space representation

Forced and unforced response

Since

$$\frac{de^{-at}x(t)}{dt} = e^{-at}\left(\dot{x}(t) - ax(t)\right),$$

it follows that

$$\frac{de^{-at}x(t)}{dt} = e^{-at}bu(t).$$

Integrating both sides

$$\int_0^t \frac{de^{-a\tau}x(\tau)}{d\tau}d\tau = e^{-at}x(t) - x(0) = \int_0^t e^{-a\tau}bu(\tau)d\tau,$$

that finally yields to

$$x(t) = e^{at}x(0) + \int_0^t e^{a(t-\tau)}bu(\tau)d\tau.$$

State space representation

Forced and unforced response

As a consequence, we have that the *forced response* is given by

$$x_f(t) = \int_0^t e^{a(t-\tau)} b u(\tau) d\tau.$$

Following the same steps for the multidimensional case, for which we notice that $e^{0t} = I$ and $\left(e^{-At}\right)^{-1} = e^{At}$, we get

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)} B u(\tau) d\tau.$$

Notice how the integral of the forced response is the *convolution integral*.

State space representation

Exercise

As an exercise, compute the *response* of the system

$$\dot{x}_1 = x_1$$
$$\dot{x}_2 = -2x_1 - x_2 + u$$

when the initial conditions are $x_1(0) = 1$ and $x_2(0) = 2$ and the input $u(t) = 10$, $\forall t \geq 0$.

Outline

State space representation

Sample and hold

Suppose that the system is sampled through a *sample and hold*, aka *zero order hold* (ZOH), i.e.,

$$u(t) = u_k \text{ for } k\Delta_t \leq t \leq (k+1)\Delta_t,$$

where Δ_t is the sampling time.
Hence, the discrete-time system dynamic is given by

$$\begin{aligned} x_{k+1} &= A_s x_k + B_s u_k \\ z_k &= C x_k \end{aligned}$$

where $x_k = x(k\Delta_t)$ and

$$A_s = e^{A\Delta_t} \text{ and } B_s = \int_0^{\Delta_t} e^{A\tau} B d\tau.$$

State space representation

Sample and hold and Euler Integration Rule

If the Euler integration rule is adopted, the dynamic becomes:

$$x_{k+1} = A_e x_k + B_e u_k$$
$$z_k = C x_k$$

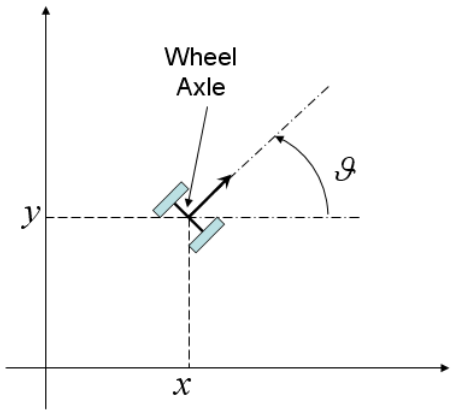
where $x_k = x(k\Delta_t)$ and

$$A_e = I + A\Delta_t \text{ and } B_e = B\Delta_t.$$

Therefore, an error of the order of Δ_t^2 is then unavoidable, since $e^{A\Delta_t} \approx I + A\Delta_t$.
This approach is sometimes called *simple derivative replacement* (SDR).

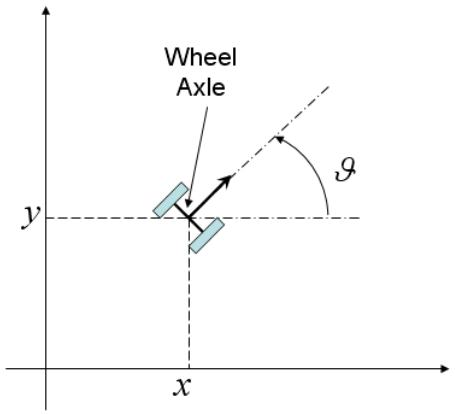
Outline

Dead Reckoning for the Unicycle



As an example of discretisation for nonlinear systems, let us consider the case of the unicycle vehicle.
Using the discretised model, it is possible to make a prediction about the position of the vehicle at each time instant Δ_t.
The input of the discrete model is given by the incremental encoders on the left and right wheels.

Dead Reckoning for the Unicycle



Let us recall the unicycle like vehicle kinematic model

$$\dot{\mathbf{q}} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}$$

Dead Reckoning for the Unicycle

- From a practical view–point, the vehicle is usually controlled using the rotational velocities (ω_l, ω_r) of the left and right wheel respectively.
- To obtain the velocities of the kinematic model, the following relations are used

$$\begin{cases} v = \frac{R}{2}(\omega_r + \omega_l) \\ \omega = \frac{R}{D}(\omega_r - \omega_l) \end{cases}$$

  where R is the wheels radius and D is the length of the wheel axle.
- Where are the encoder measurements in the model? And, how to use them?

Dead Reckoning for the Unicycle

- Since the encoder values represent the angular position η of each wheel and assuming realistically that we sample the encoders with a period of Δt, we can apply the Euler integration to approximate the time derivatives, i.e.,

$$\begin{cases} \omega_r \approx \frac{\eta_r(t+\Delta t)-\eta_r(t)}{\Delta t} \\ \omega_l \approx \frac{\eta_l(t+\Delta t)-\eta_l(t)}{\Delta t} \end{cases} \Rightarrow \begin{cases} \omega_r \Delta t \approx \eta_r(t+\Delta t)-\eta_r(t) \\ \omega_l \Delta t \approx \eta_l(t+\Delta t)-\eta_l(t) \end{cases}$$

- Hence, we have

$$\begin{cases} v\Delta t \approx \frac{R}{2}(\eta_r(t+\Delta t)-\eta_r(t)+\eta_l(t+\Delta t)-\eta_l(t)) \\ \omega\Delta t \approx \frac{R}{D}(\eta_r(t+\Delta t)-\eta_r(t)-\eta_l(t+\Delta t)+\eta_l(t)) \end{cases}$$

Dead Reckoning for the Unicycle

- Similarly and finally

$$\dot{\mathbf{q}}\Delta t = \begin{bmatrix} \dot{x}\Delta t \\ \dot{y}\Delta t \\ \dot{\theta}\Delta t \end{bmatrix} = \begin{bmatrix} x(t+\Delta t) - x(t) \\ y(t+\Delta t) - y(t) \\ \theta(t+\Delta t) - \theta(t) \end{bmatrix} = \begin{bmatrix} \cos\theta(t) & 0 \\ \sin\theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v\Delta t \\ \omega\Delta t \end{bmatrix}$$

from which we have the sampled, reconstructed dynamic given by

$$\begin{bmatrix} x(t+\Delta t) \\ y(t+\Delta t) \\ \theta(t+\Delta t) \end{bmatrix} = \begin{bmatrix} x(t) + \cos(\theta(t))v\Delta t \\ y(t) + \sin(\theta(t))v\Delta t \\ \theta(t) + \omega\Delta t \end{bmatrix}$$

Outline

Dead Reckoning – Remarks

- The estimator here proposed reconstruct the actual position of the robot given the *story* of the encoder values.
- It is trivial that such an algorithm can be successfully implemented whenever the initial condition for the robot, i.e., $[x(0),\, y(0),\, \theta(0)]$, and for the encoders, i.e., $[\eta_r(0),\, \eta_l(0)]$, are known or chosen by the designer.
- Notice that the wheel encoders are the only sources of information, hence the high inaccuracy of the dead–reckoning for robot position reconstruction.
- Adding more sensors, all the additional data can be coherently fused together using *Extended Kalman Filters* to get a finer estimate of the state $\mathbf{q}$.

Outline

Inter sample model

Sometimes it is useful to have the dynamic of the system in the inter sample, i.e., with a time that is more fine grained than the sampling time Δ_t adopted in the S&H.
This is very convenient when, for example, the time of arrival of messages, e.g., *control signals* $u(k)$, are ruled by an external random process.
For example, this is what happens when the control signals are computed by a platform with unpredictable time of execution or when there is a network in between.
Hence, to have a picture of the inter sampling behaviour, it is needed just to sample the system with a *finer* sampling time δ_t, i.e., $\Delta_t = n\delta_t$, with $n > 1$.
To recover the original dynamic, it is sufficient to compute the forced and unforced response of the system for the *discrete time*.

Event-based Sampling

It is possible to sample the system using a different approach for sampling, i.e., modifying the strategy from the classic S&H *time based* to an *event based* strategy.
We already know that this is very advantageous for *networked* control systems.
This is also of relevance for *soft real-time* systems.
The basic idea can be restated by simply saying that the sampling changes from the *Riemann sampling* (time-based) to the *Lebesgue sampling* (event-based).

Event-based Sampling

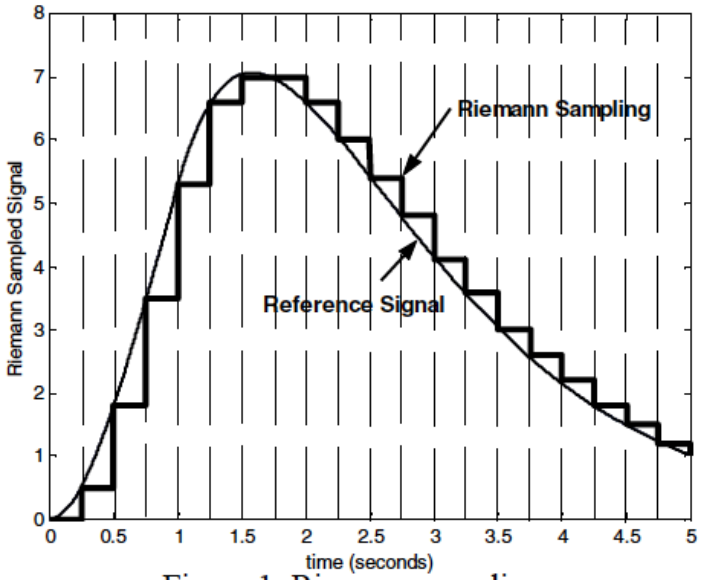


Figure 1. Riemann sampling.

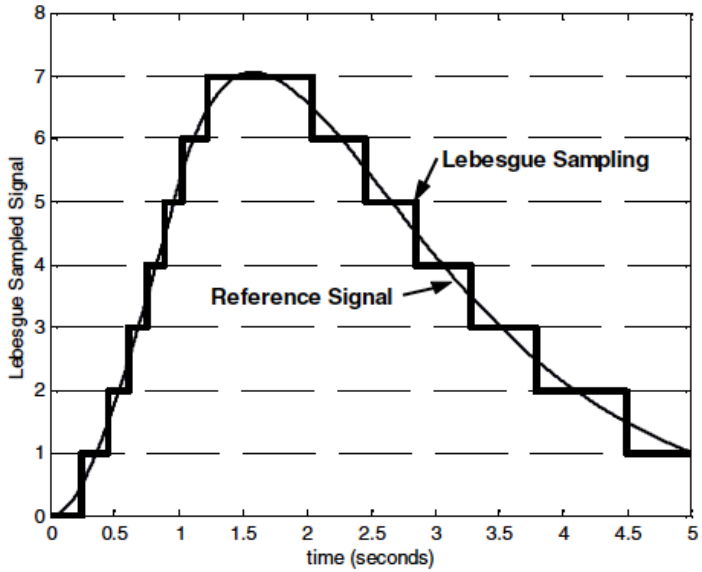


Figure 2. Lebesgue sampling.

Figure: Riemann (Fig. 1) and Lebesgue (Fig. 2) samplers (Roy McCann, Anil Kumar Gunda, Suchit Reddy Damugatla, "Improved Operation of Networked Control Systems using Lebesgue Sampling", IAS 2004).

Noise

Unfortunately, the system we are dealing with are *stochastic* linear systems, hence the dynamic will be given by

$$\dot{x}(t) = Ax(t) + Bu(t) + G\tilde{\nu}(t),$$

where $\tilde{\nu}(t)$ is supposed to be white with zero mean and possibly time varying variance $V(t)$, i.e.,

$$\mathsf{E}\left\{\tilde{\nu}(t)\right\} = 0 \text{ and } \mathsf{E}\left\{\tilde{\nu}(t)\tilde{\nu}(\tau)^T\right\} = V(t)\delta(t-\tau),$$

where $\delta(\cdot)$ is the standard *Dirac delta function*.

Noise

Due to the discretisation, the system is rewritten with the difference equation

$$x(k+1) = A_D x(k) + B_D u(K) + \nu(k),$$

being $\nu(k)$ is the discretised version of the continuous noise $\tilde{\nu}(t)$ weighted by the matrix G.
Following the same arguments of the previous analysis, we may rewrite

$$x(t_1) = e^{A(t_1-t_0)}x(t_0) + \int_{t_0}^{t_1} e^{A(t_1-\tau)}[B(\tau)u(\tau) + G(\tau)\tilde{\nu}(\tau)]d\tau,$$

from which the *Markov property* is further highlighted, i.e.,

$$f(x(t_1)|x_{(-\infty,t_0)}, u_{(t_0,t_1)}, \tilde{\nu}_{(t_0,t_1)}) = f(x(t_1)|x(t_0), u_{(t_0,t_1)}, \tilde{\nu}_{(t_0,t_1)}).$$

Noise

From the previous analysis and the linearity of the integral operator, it follows immediately that

$$\nu(k) = \int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau)} G(\tau)\tilde{\nu}(\tau)d\tau.$$

Let us derive the *stochastic description* of this new discrete time random variable.
For the mean value, it is immediate (due to the linearity of the integral operator) to show that

$$\mathsf{E}\{\nu(k)\} = 0.$$

Noise

For the variance, the analysis is a little bit more complicated

$$\mathsf{E}\left\{\nu(k)\nu(j)^T\right\} =$$

$$= \mathsf{E}\left\{\int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau_k)}G(\tau_k)\tilde{\nu}(\tau_k)d\tau_k \int_{t_j}^{t_{j+1}} \tilde{\nu}(\tau_j)^T G(\tau_j)^T e^{A(t_{j+1}-\tau_j)^T}d\tau_j\right\} =$$

$$= \int_{t_k}^{t_{k+1}}\int_{t_j}^{t_{j+1}} e^{A(t_{k+1}-\tau_k)}G(\tau_k)\mathsf{E}\left\{\tilde{\nu}(\tau_k)\tilde{\nu}(\tau_j)^T\right\}G(\tau_j)^T e^{A(t_{j+1}-\tau_j)^T}d\tau_j d\tau_k =$$

$$= \int_{t_k}^{t_{k+1}}\int_{t_j}^{t_{j+1}} e^{A(t_{k+1}-\tau_k)}G(\tau_k)V(\tau_k)\delta(\tau_k-\tau_j)G(\tau_j)^T e^{A(t_{j+1}-\tau_j)^T}d\tau_j d\tau_k$$

Noise

Hence

$$\mathsf{E}\left\{\nu(k)\nu(j)^T\right\} =$$
$$= \int_{t_k}^{t_{k+1}} e^{A(t_{k+1}-\tau_k)} G(\tau_k) V(\tau_k) G(\tau_k)^T e^{A(t_{k+1}-\tau_k)^T} d\tau_k \delta_{k,j}$$

where $\delta_{k,j}$ is the *Kronecker delta* function.
In practice, the larger is the sampling period, the larger would be the variance of the *discrete white noise* injected in the discrete time system.