Vlad MUNTEANU

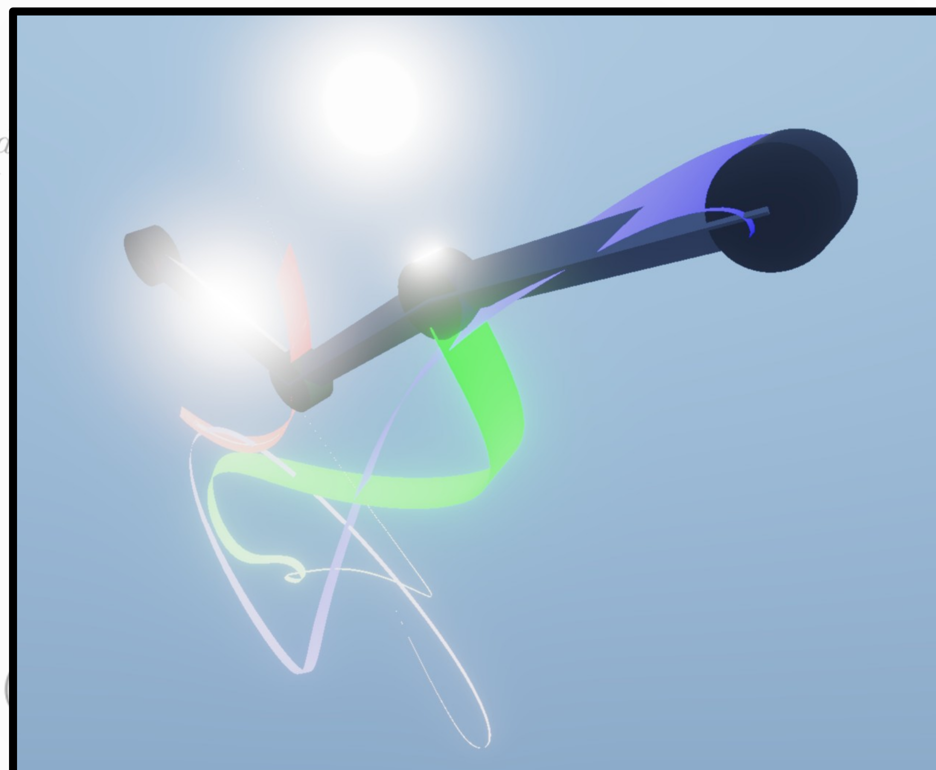
Simulation physique de solides indéformables

$K_i^o = \begin{pmatrix} \iint_{P \in V} \rho_i(P)(y^2+z^2) \\ \iint_{P \in V} -\rho_i(P) xy \, dV \\ \iint_{P \in V} -\rho_i(P) xz \, dV \end{pmatrix}$

$\cdot \dot{q} = {}^t F_C \dot{q} = 0$

et $J(q)\dot{q} = 0$

$M\ddot{q} = F$



$_i = R(\theta_i) K_i^o R(\theta_i)^{-1}$

$f(t + \Delta t) \approx f(t) + \Delta t f'($

$= k^2 = ijk = -1$

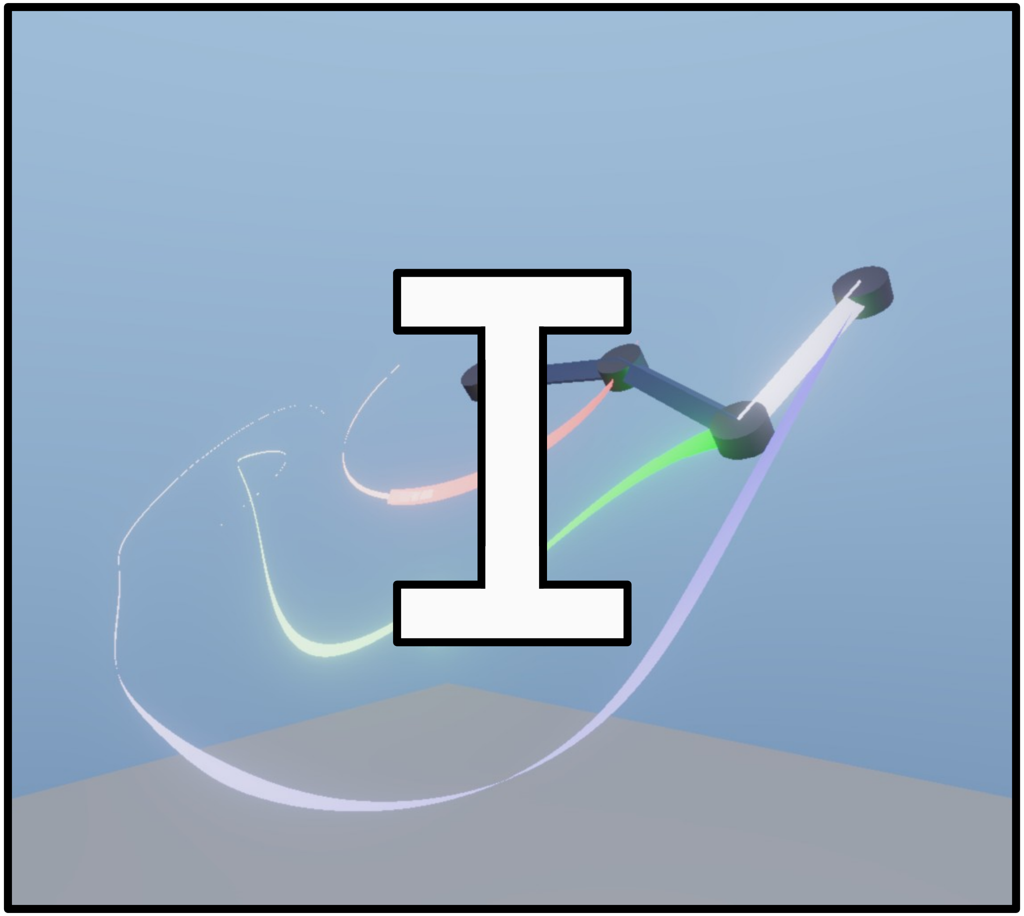
Problématique :

Comment produire une simulation physique de solides indéformables à la fois rapide et précise ?

Sommaire

I. Objectifs
II. Cadre physique
III. Implémentation informatique
IV. Résultats

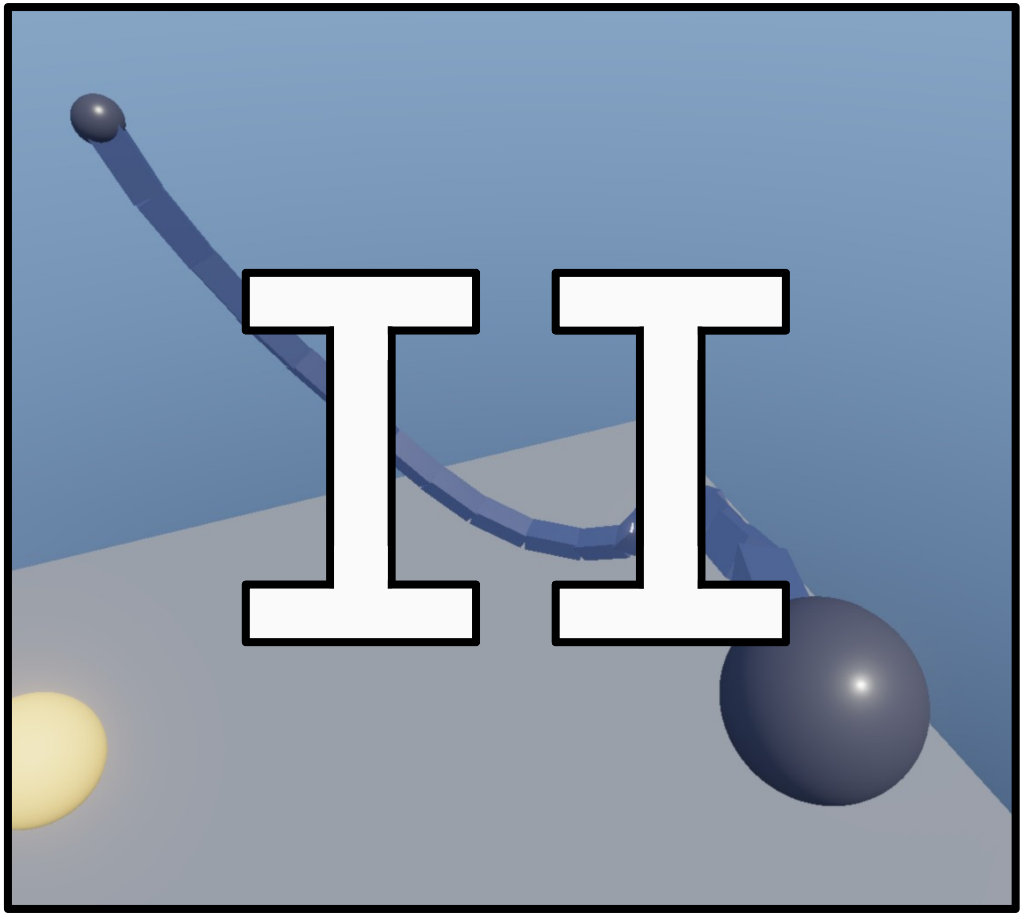
Objectifs



I. Objectifs

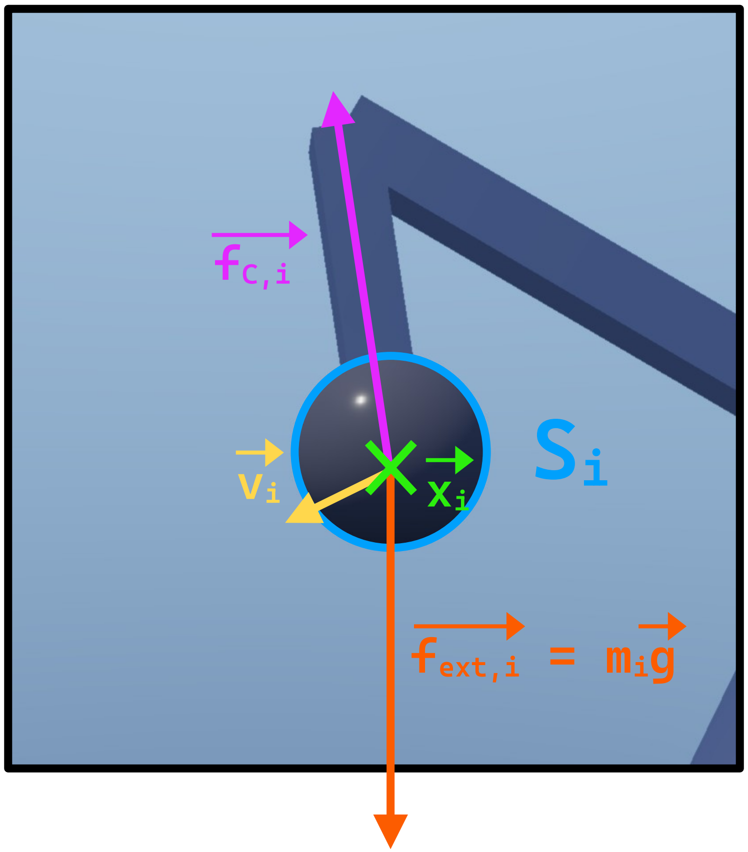
- Physiquement correct
- Précis
- Temps réel

Cadre physique



II. Cadre Physique

Représentation vectorielle : **solide S_i**



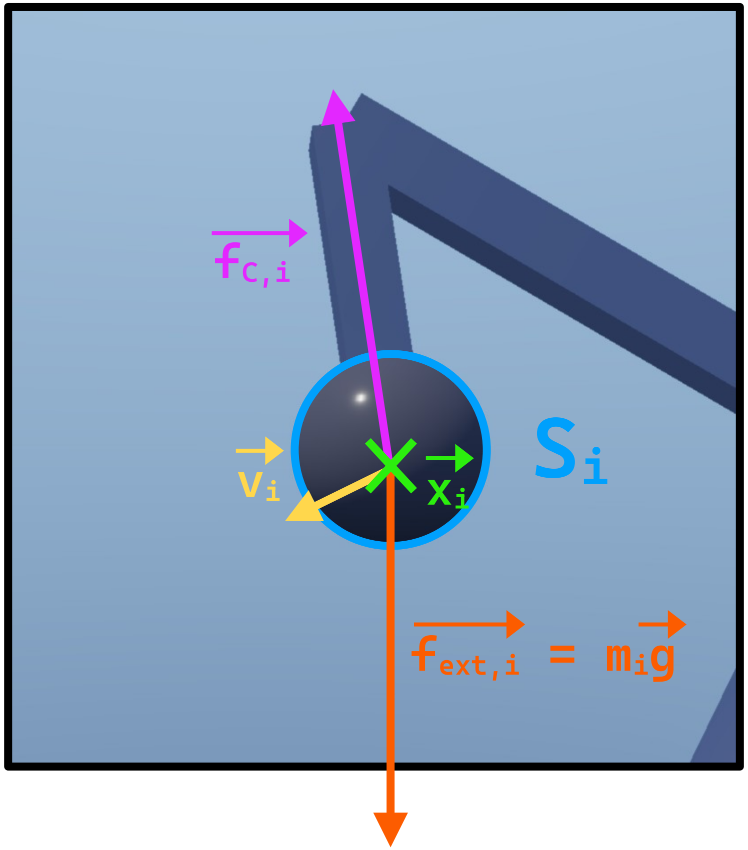
$$q_i = \begin{pmatrix} x_i \\ \theta_i \end{pmatrix} \quad \dot{q}_i = \begin{pmatrix} \dot{x}_i \\ \dot{\theta}_i \end{pmatrix} \quad \ddot{q}_i = \begin{pmatrix} \ddot{x}_i \\ \ddot{\theta}_i \end{pmatrix}$$

$$m_i \qquad K_i = R(\theta_i) K_i^o R(\theta_i)^{-1}$$

$$M_i = \begin{pmatrix} m_i I_3 & 0_3 \\ 0_3 & K_i \end{pmatrix}$$

II. Cadre Physique

Représentation vectorielle : **solide S_i**

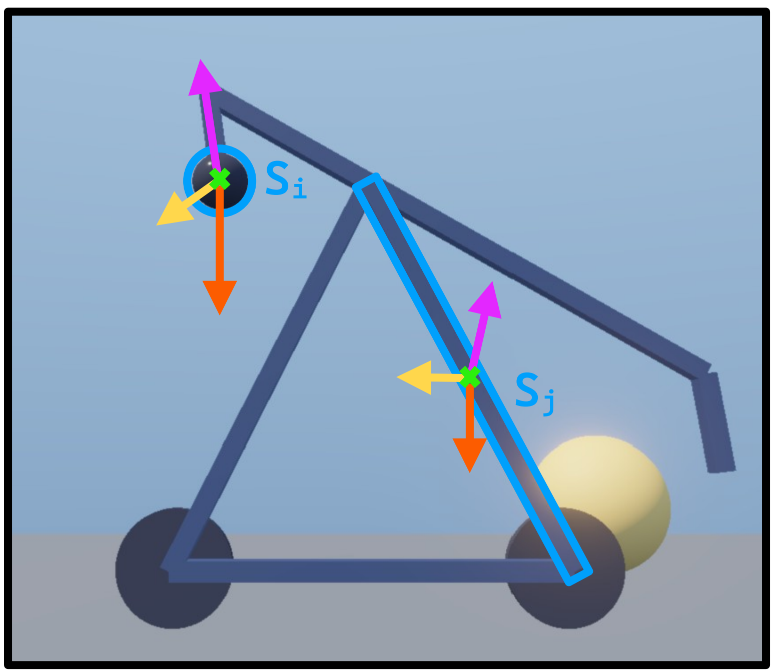


$$F_i = F_{ext,i} + F_{C,i} = \begin{pmatrix} f_{ext,i} \\ \tau_{ext,i} \end{pmatrix} + \begin{pmatrix} f_{C,i} \\ \tau_{C,i} \end{pmatrix}$$

$$M_i \ddot{q}_i = F_i \Leftrightarrow \begin{cases} m_i a_i = f_{ext,i} + f_{C,i} \\ K_i \alpha_i = \tau_{ext,i} + \tau_{C,i} \end{cases}$$

II. Cadre Physique

Représentation vectorielle : **système $\boldsymbol{S}$** $= \{S_i\}_{i\in[1,n]}$



$$q = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix} = \begin{pmatrix} x_1 \\ \theta_1 \\ \vdots \\ x_n \\ \theta_n \end{pmatrix} \qquad M = \begin{pmatrix} M_1 & 0 & \cdots & 0 \\ 0 & M_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & M_n \end{pmatrix}$$

$$F = F_{ext} + F_C = \begin{pmatrix} F_{ext,1} \\ F_{ext,2} \\ \vdots \\ F_{ext,n} \end{pmatrix} + \begin{pmatrix} F_{C,1} \\ F_{C,2} \\ \vdots \\ F_{C,n} \end{pmatrix} = \begin{pmatrix} f_{ext,1} \\ \tau_{ext,1} \\ \vdots \\ f_{ext,n} \\ \tau_{ext,n} \end{pmatrix} + \begin{pmatrix} f_{C,1} \\ \tau_{C,1} \\ \vdots \\ f_{C,n} \\ \tau_{C,n} \end{pmatrix}$$

$$M\ddot{q} = F = F_{ext} + F_C \Leftrightarrow \begin{cases} M_1\ddot{q}_1 & = F_{ext,1} + F_{C,1} \\ & \vdots \\ M_n\ddot{q}_n & = F_{ext,n} + F_{C,n} \end{cases}$$

II. Cadre Physique

Concept de **contrainte**

$$C_{l_j,j} : \mathcal{Q}(S) \to \mathbb{R}^{l_j}$$

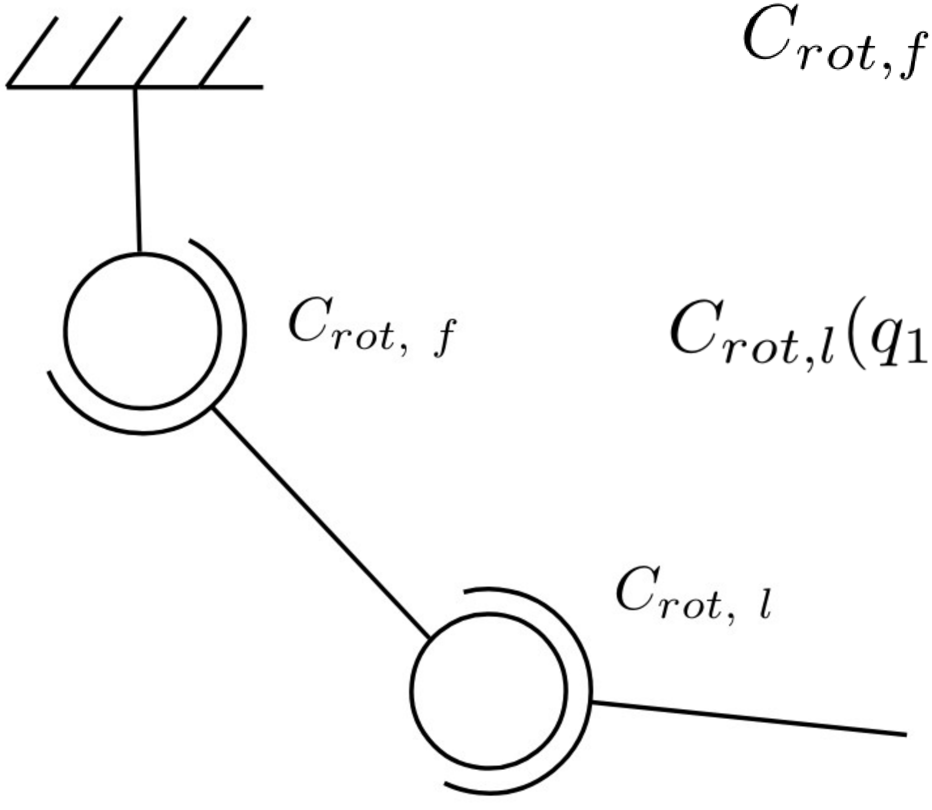
$$C_{l_j,j}(q(t)) = 0$$

$$C = \begin{pmatrix} C_{l_1,1} \\ \vdots \\ C_{l_m,m} \end{pmatrix}$$

$$\frac{\mathrm{d}C}{\mathrm{d}t}(q) = \frac{\partial C}{\partial q}(q)\frac{\mathrm{d}q}{\mathrm{d}t} = J(q)\dot{q} \text{ avec } J = \frac{\partial C}{\partial q}$$

II. Cadre Physique

Concept de **contrainte**



$$C_{rot,f}(q_1) = x_1 + {}^tR(\theta_1)o_1R(\theta_1) - p$$

$$\begin{aligned} C_{rot,l}(q_1, q_2) = x_1 + {}^tR(\theta_1)o_1R(\theta_1) \\ - x_2 - {}^tR(\theta_2)o_2R(\theta_2) \end{aligned}$$

II. Cadre Physique

Formule de $\boldsymbol{F_c}$ d'inconnue $\boldsymbol{\lambda}$

(…)

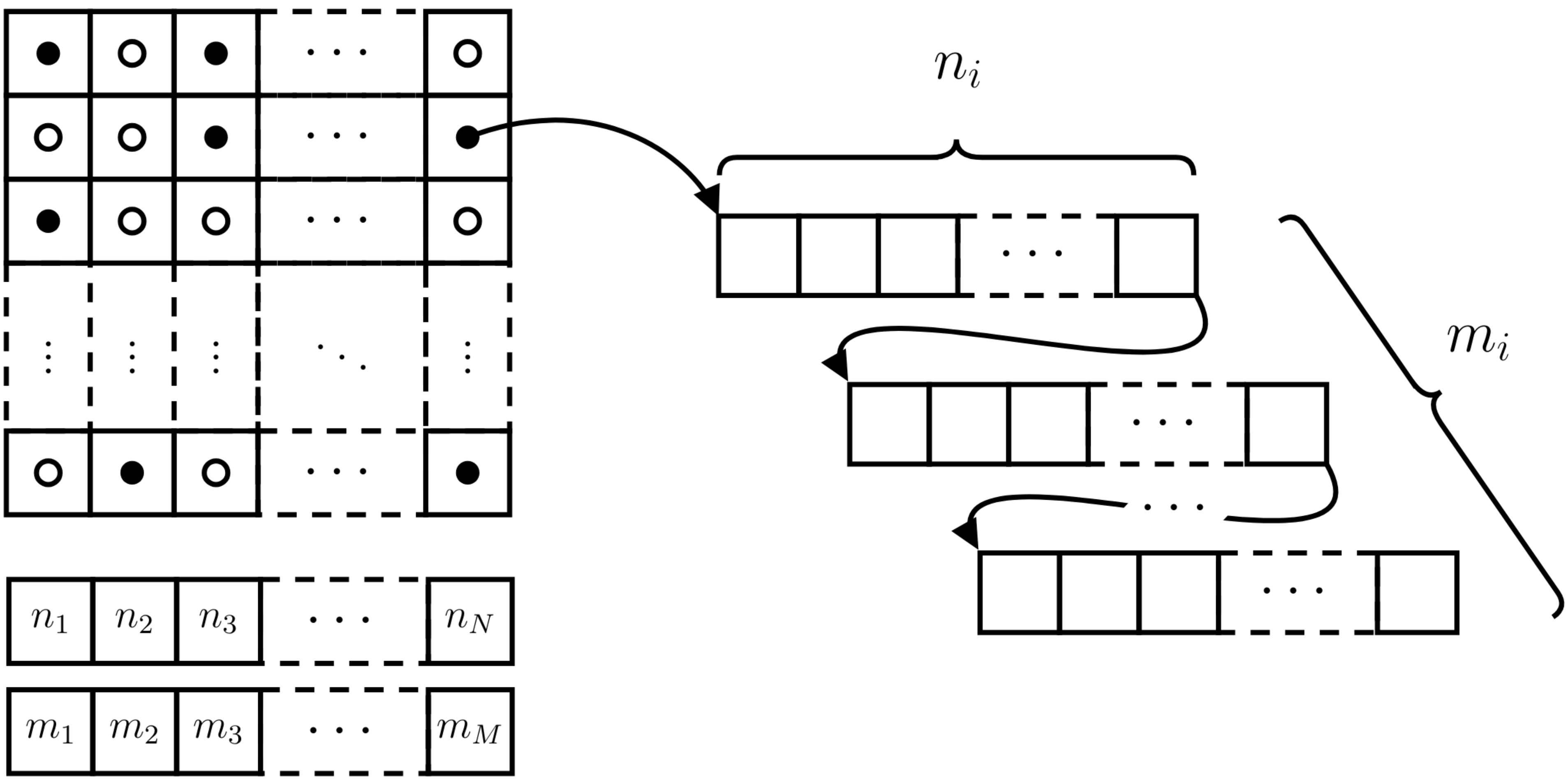
$$J(q)M^{-1}\,{}^{t}J(q)\lambda = -\dot{J}(q)\dot{q} - J(q)M^{-1}F_{ext}$$

$${}^{t}J(q)\lambda = F_C$$

Implémentation informatique

III. Implémentation informatique

Structures : **matrices par blocs**



III. Implémentation informatique

Structures : **quaternions**

$$i^2 = j^2 = k^2 = ijk = -1$$

$$\mathbb{H} = \mathrm{Vect}(1, i, j, k) \simeq \mathbb{R}^4$$

$$U(1, \mathbb{H}) = \{q \in \mathbb{H},\ \|q\|_2 = 1\} \simeq SO_3(\mathbb{R})$$

$$\text{Rotation de } v = \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix} \in \mathbb{R}^3 \text{ par } q = \begin{pmatrix} x_q \\ y_q \\ z_q \\ w_q \end{pmatrix} \in U(1, \mathbb{H}) :$$

$$v_q := 0 + \mathbf{i}x_v + \mathbf{j}y_v + \mathbf{k}z_v$$

$$v'_q := q v_q q^{-1} = q v_q \overline{q}$$

$$v' := \begin{pmatrix} y_{v'_q} \\ z_{v'_q} \\ w_{v'_q} \end{pmatrix}$$

III. Implémentation informatique

Algorithmes : **intégration**

Méthode d'Euler $\longrightarrow$ Runge-Kutta 4 (RK4)

Calcul vélocités à $t + \Delta t$:

$$\dot{q}(t+\Delta t) = \ddot{q}(t) + M^{-1} F \Delta t$$

$$\Leftrightarrow$$

$$\forall i \in [\![1, n]\!], \begin{cases} v_i(t+\Delta t) = v_i(t) + m_i f_i(t+\Delta t)\Delta t \\ \omega_i(t+\Delta t) = \omega_i(t) + K_i(t)\tau_i(t+\Delta t)\Delta t \end{cases}$$

Calcul positions à $t + \Delta t$:

$$\forall i \in [\![1, n]\!], \begin{cases} x_i(t+\Delta t) = x_i(t) + v_i(t+\Delta t)\Delta t \\ \theta_i(t+\Delta t) = \theta_i(t) + \frac{1}{2}\omega_i(t+\Delta t) \times \theta_i(t)\Delta t \end{cases}$$

III. Implémentation informatique

Algorithmes : **système linéaire**

Gauss-Seidel $\longrightarrow$ Bi-Conjugate Gradients Stabilized
(BiCGSTAB)

$$A \in \mathcal{M}_n(\mathbb{R}),\, b \in \mathbb{R}^n.\ \text{Trouver } x \in \mathbb{R}^n \text{ tel que } Ax = b$$

ALG3 / MC64
(Conditionnement)

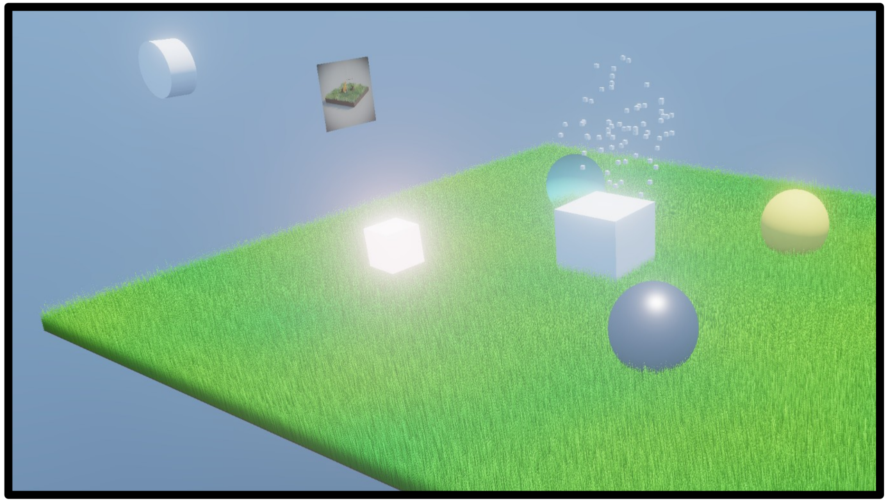
III. Implémentation informatique

Interface utilisateur, librairies

SupSy Librairies (SL)

GLFW + OpenGL

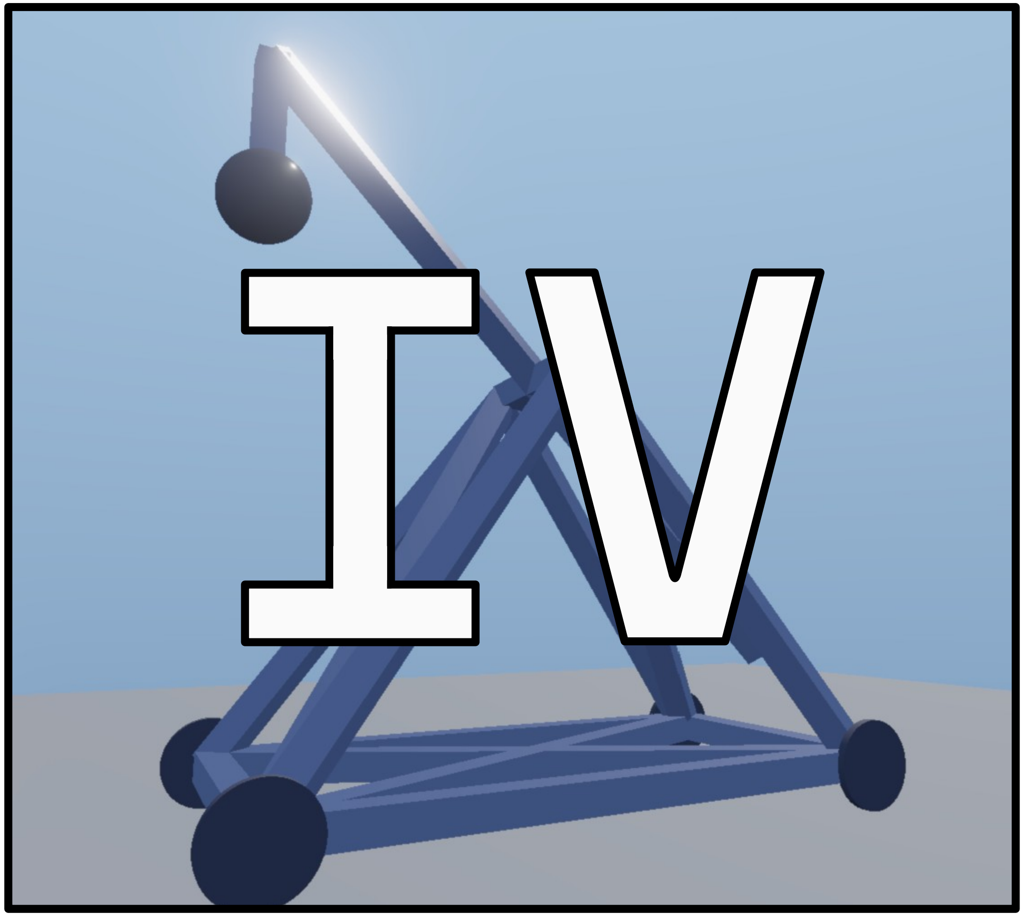
SupSy Game Engine (SGE)



```c
#include <SupSy/SGE.h>
#include <SupSy/SGE/builtin/extData/freeCam.h>

int main() {
    initializeApp("Small App");
    freeCam_addDefault((vec3*)&vec3_zero, (quat*)&quat_identity, 60, false);

    while (!appShouldClose()) {
        startFrameUpdate();

        sceneUpdate(APP->scene);
        RERenderScene(APP->scene, REGetOutputFB(APP->renderEnvironment));
        blitToScreenFB(REGetOutputFB(APP->renderEnvironment));

        endFrameUpdate();
    }
}
```

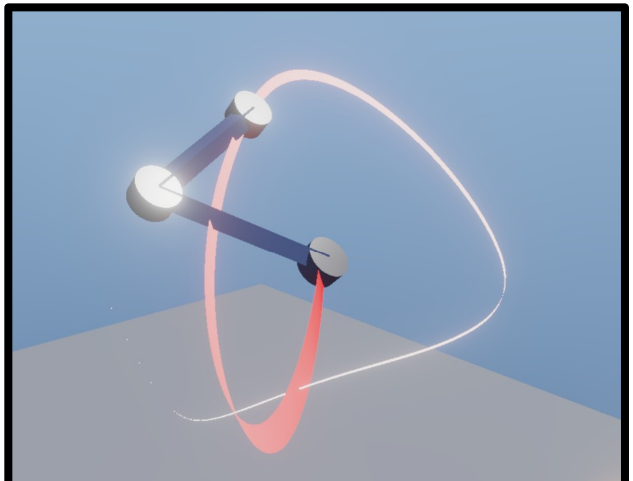
- SL
  - maths
    - constants.h M
    - math.c U
    - math.h U
    - matrix.c M
    - matrix.h M
    - quaternion.c M
    - quaternion.h M
    - vector.c M
    - vector.h M
  - utils
    - arenaAlloc.c
    - arenaAlloc.h
    - array.c U
    - array.h U
    - debug.c M
    - debug.h M
    - imageImporter.c
    - imageImporter.h
    - inout.c
    - inout.h M
    - list.c U
    - list.h U

Résultats



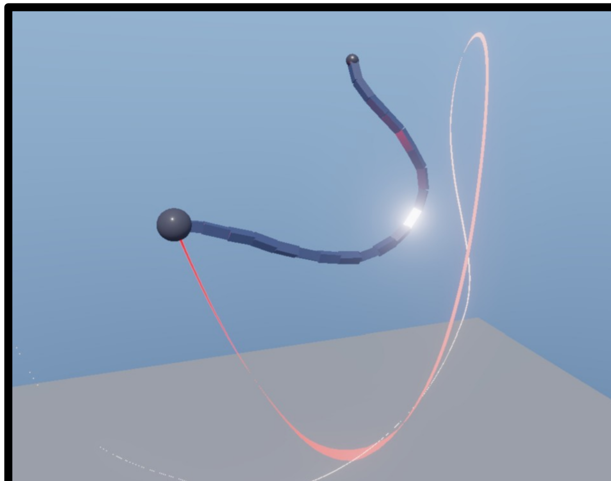
IV. Résultat

Double Pendule 45°



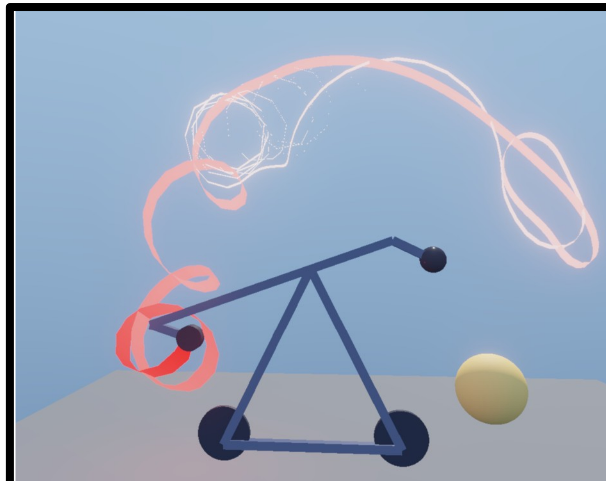
```
Times [140]:
        Inverse Inertia -> 1.527600ms, 1.710572ms
        J and dJ -> 3.873300ms, 4.386572ms
        JW -> 1.946700ms, 1.886380ms
        Right member -> 4.871400ms, 5.337228ms
        Left member -> 16.973000ms, 18.641968ms
        System equilibrating -> 57.458000ms, 66.411700ms
        System resolution -> 309.783600ms, 314.220152ms
        Force -> 2.555400ms, 2.807116ms
        Position update -> 1.591300ms, 1.771240ms
        Velocity update -> 0.530800ms, 0.514836ms
        ==> True total: 401.111100ms, 417.687764ms
Energy: 14672.401657 J
Total: 0.530800ms, 0.514836ms  --  At: 200.698717s
```

Corde 20 segments



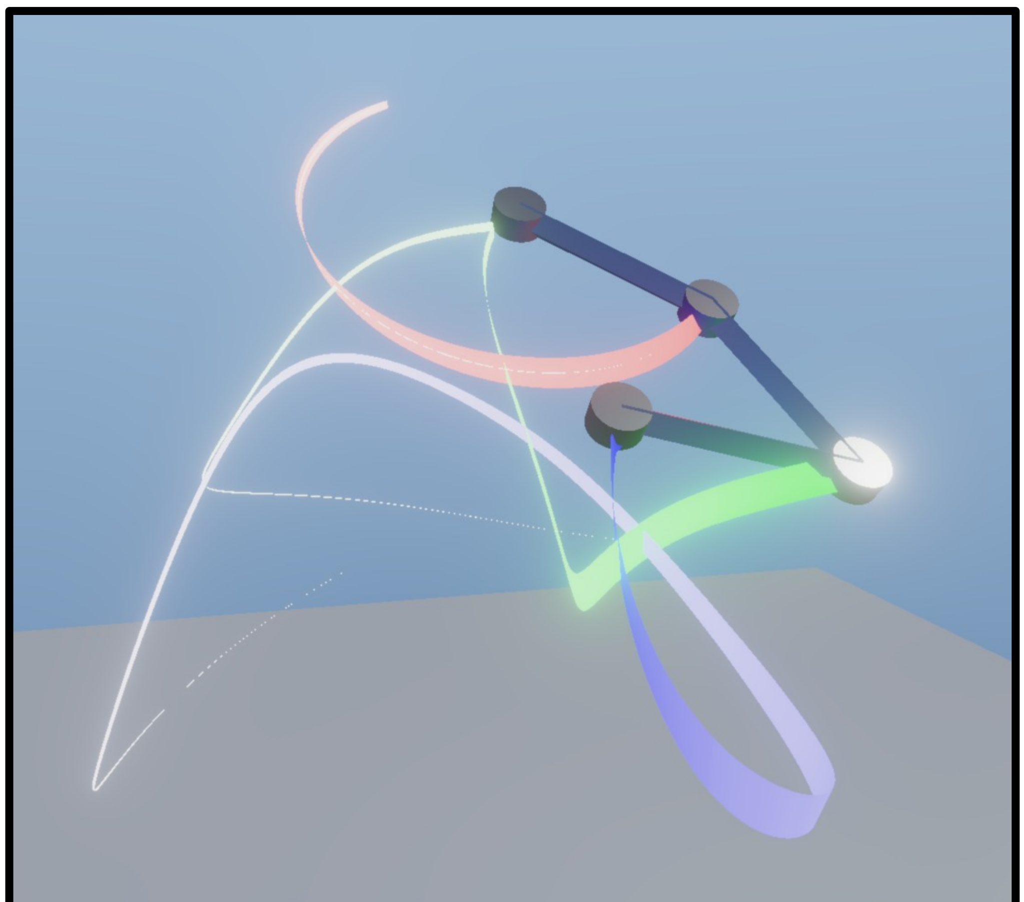
```
Times [160]:
        Inverse Inertia -> 6.542200ms, 6.486256ms
        J and dJ -> 16.400100ms, 16.828532ms
        JW -> 13.362000ms, 12.270132ms
        Right member -> 22.640900ms, 22.178256ms
        Left member -> 133.400800ms, 118.849896ms
        System equilibrating -> 151.426500ms, 149.185368m
        System resolution -> 620.076700ms, 607.704968ms
        Force -> 15.896200ms, 13.900476ms
        Position update -> 7.078800ms, 7.302660ms
        Velocity update -> 1.507700ms, 1.456248ms
        ==> True total: 988.331900ms, 956.162792ms
Energy: 1803.084927 J
Total: 1.507700ms, 1.456248ms  --  At: 177.098959s
```

Trébuchet 2D



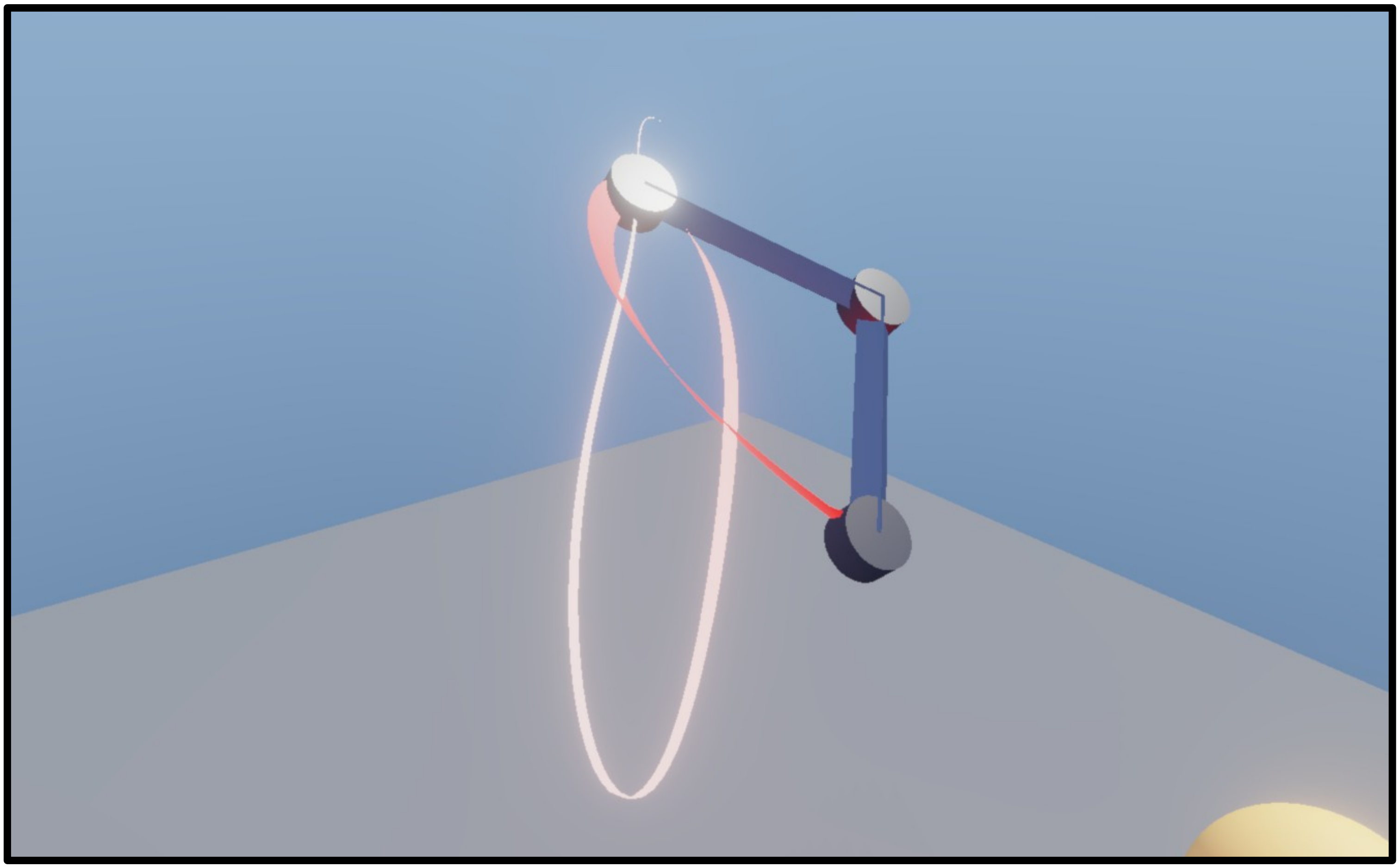
```
Times [143]:
        Inverse Inertia -> 0.000000ms, 0.000000ms
        J and dJ -> 57.867300ms, 53.812748ms
        JW -> 19.308200ms, 18.654800ms
        Right member -> 23.092600ms, 21.720828ms
        Left member -> 96.795800ms, 91.292516ms
        System resolution -> 278.468600ms, 270.097928ms
        Force -> 11.836100ms, 11.425192ms
        Position update -> 1.602500ms, 1.581064ms
        Velocity update -> 1.717300ms, 1.680744ms
        ==> True total: 490.688400ms, 470.265820ms
Energy: 4102.868359 J
Total: 1003.470400ms, 1000.716100ms  --  At: 183.140110s
```

Conclusion



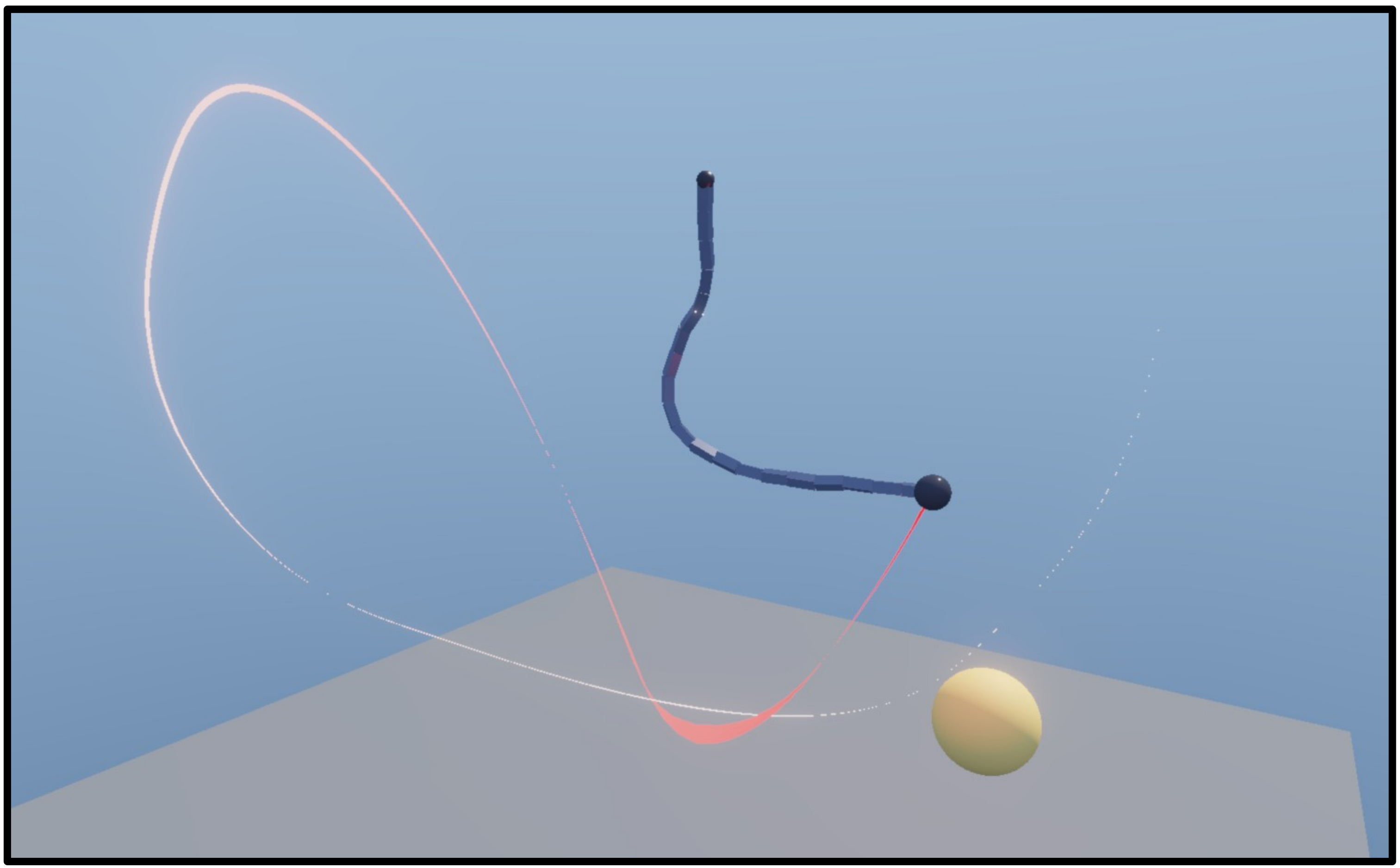
Annexe

Captures



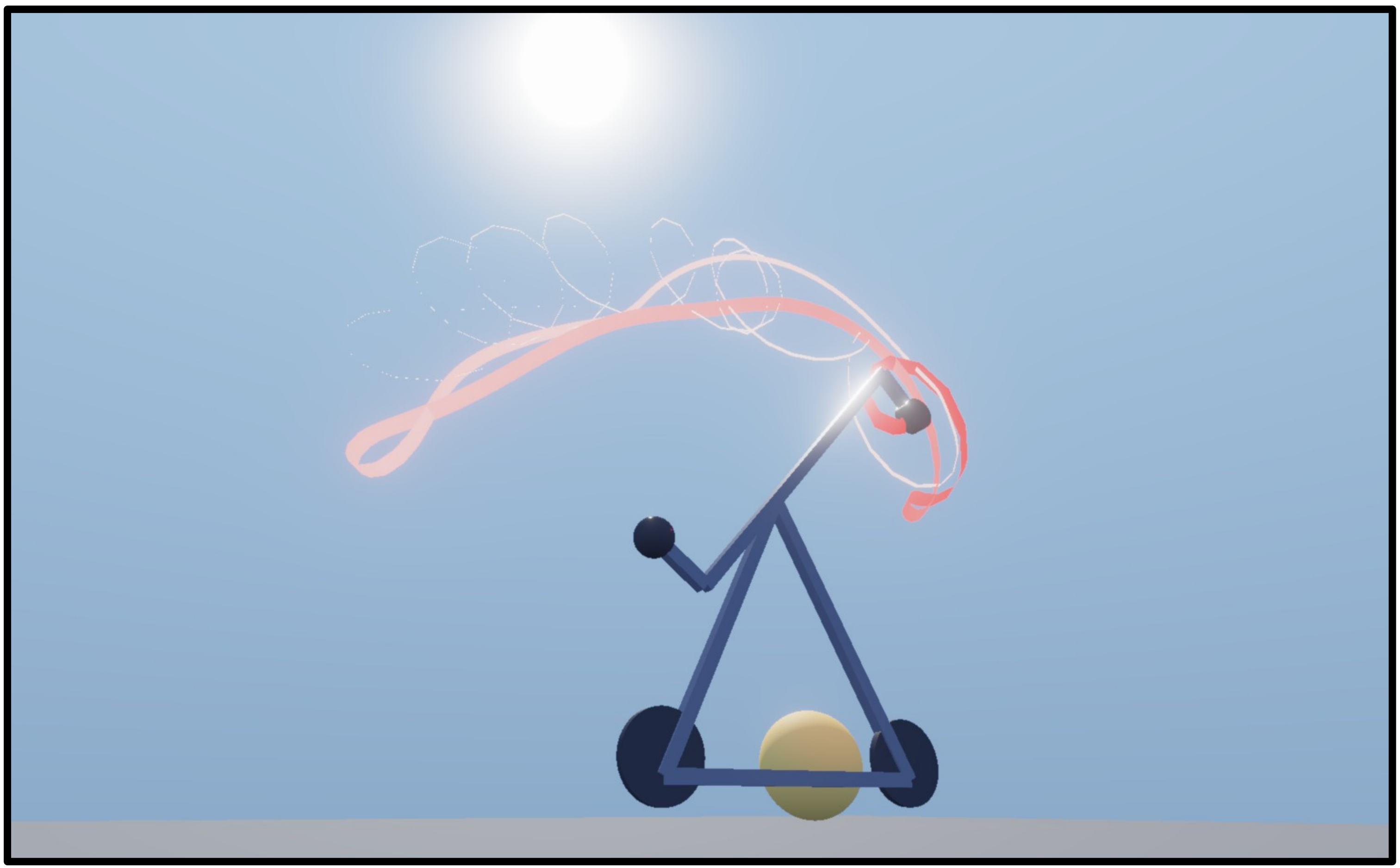
Annexe

Captures



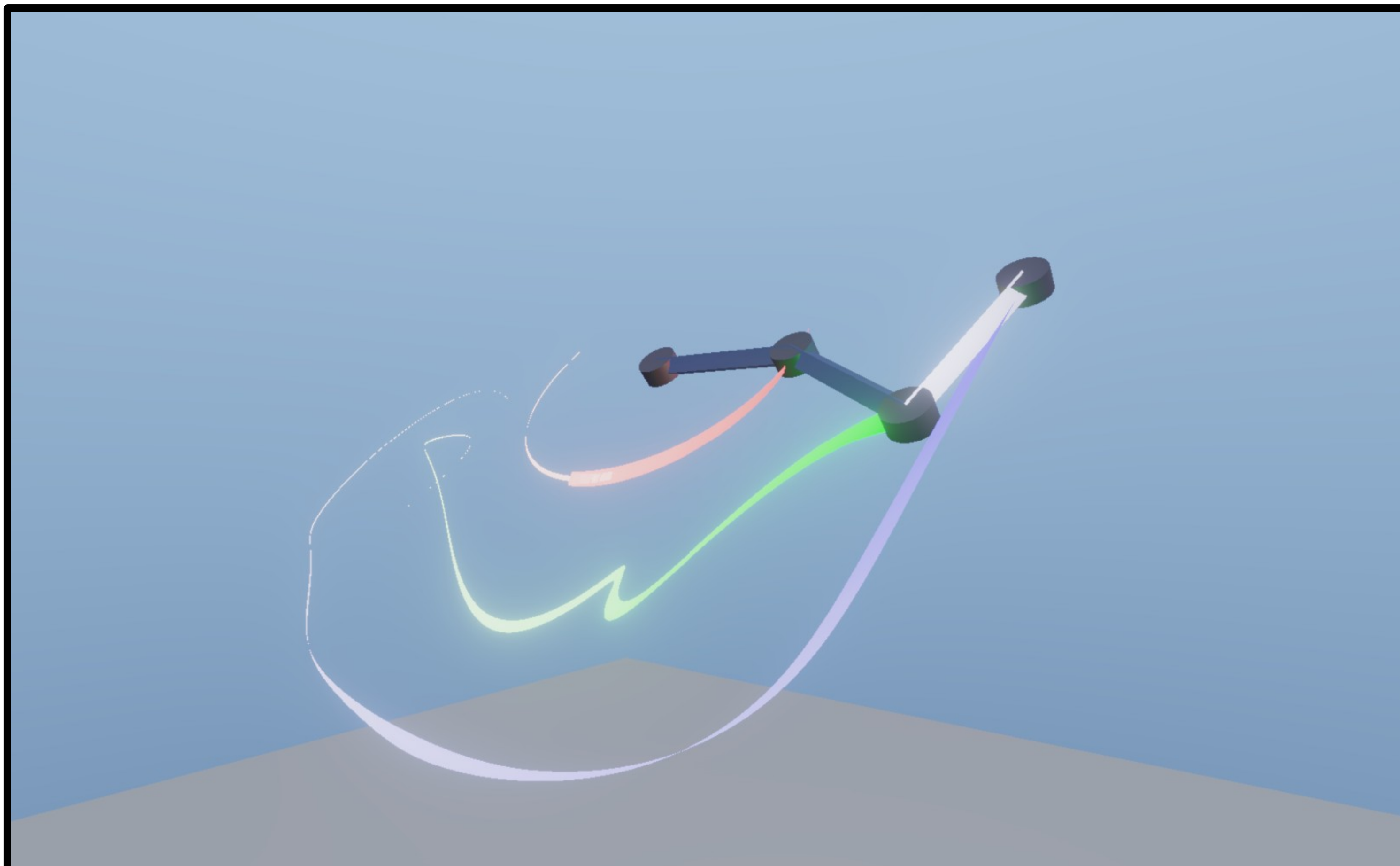
Annexe

Captures



Annexe

Captures



Annexe

Explications détaillées : **formule de F_C**

$$C(q) = 0$$

$$\frac{\mathrm{d}C}{\mathrm{d}t}(q) = J(q)\dot{q} = 0$$

$$\frac{\mathrm{d}^2C}{\mathrm{d}t^2}(q) = \dot{J}(q)\dot{q} + J(q)\ddot{q} = 0$$

$$F_C \cdot \dot{q} = {}^tF_C\dot{q} = 0$$

$$\text{et } J(q)\dot{q} = 0$$

$${}^tJ(q)\lambda = F_C$$

$$\dot{J}(q)\dot{q} + J(q)\ddot{q} = \dot{J}(q)\dot{q} + J(q)M^{-1}(F_{ext} + F_C) = 0$$

$$\boxed{J(q)M^{-1}{}^tJ(q)\lambda = -\dot{J}(q)\dot{q} - J(q)M^{-1}F_{ext}}$$

Annexe

Explications détaillées : **inversibilité de *M***

$$M \in GL_{6n}(\mathbb{R}) \Leftrightarrow \forall i \in [\![1,n]\!],\ M_i = \begin{pmatrix} m_i I_3 & 0_3 \\ 0_3 & K_i \end{pmatrix} \in GL_6(\mathbb{R})$$
$$\Leftrightarrow \forall i \in [\![1,n]\!],\ m_i \neq 0 \wedge K_i = R(\theta_i) K_i^o R(\theta_i)^{-1} \in GL_3(\mathbb{R})$$
$$\Leftrightarrow \forall i \in [\![1,n]\!],\ m_i \neq 0 \wedge K_i^o \in GL_3(\mathbb{R})$$

$$K_i^o = \begin{pmatrix} \iint_{P\in V} \rho_i(P)(y^2+z^2)\,dV & \iint_{P\in V} -\rho_i(P)xy\,dV & \iint_{P\in V} -\rho_i(P)xz\,dV \\ \iint_{P\in V} -\rho_i(P)xy\,dV & \iint_{P\in V} \rho_i(P)(x^2+z^2)\,dV & \iint_{P\in V} -\rho_i(P)yz\,dV \\ \iint_{P\in V} -\rho_i(P)xz\,dV & \iint_{P\in V} -\rho_i(P)yz\,dV & \iint_{P\in V} \rho_i(P)(x^2+y^2)\,dV \end{pmatrix}$$

$$\iint_{P\in V} \rho_i(P)(y^2+z^2)\,dV = I_{Ox} > 0$$
$$\iint_{P\in V} \rho_i(P)(x^2+z^2)\,dV = I_{Oy} > 0$$
$$\iint_{P\in V} \rho_i(P)(y^2+z^2)\,dV = I_{Oz} > 0$$

$$\det(K_i^o) = I_{Ox} I_{Oy} I_{Oz} > 0$$

Annexe

Explications détaillées : **existence de λ**

On considère que la contrainte est toujours satisfaite, donc que :

$$\forall q \in \mathcal{Q}(S),\ \tfrac{dC}{dt}(q) = J(q)\dot{q} = 0$$

Autrement dit :

$$\{\dot{q} \mid q \in \mathcal{Q}(S)\} = \ker(J(q))$$

De plus, par application du principe de moindre action, on sait que F_C ne travaille pas donc :

$$F_C \cdot \dot{q} = 0$$

F_C est donc orthogonal à tout vecteur de $\ker(J(q))$:

$$F_C \in \ker(J(q))^{\perp}$$
$$\Leftrightarrow$$
$$F_C \in \mathrm{Im}({}^tJ(q))$$

F_C admet un antécédant par ${}^tJ(q)$, il existe λ tel que $F_C = {}^tJ(q)\lambda$

Annexe

Explications détaillées : **conditionnement**

Conditionnement :
$A \in \mathcal{M}_n(\mathbb{R})$, $b \in \mathbb{R}^n$. Trouver $x \in \mathbb{R}^n$ tel que $Ax = b$

$\|\cdot\|_s$ norme suborndonnée respectivement à la norme de Frobenius :

$$(A + \delta A)(x + \delta x) = b \qquad \mathcal{K}(A) = \frac{1}{n}\|A\|_s \left\|A^{-1}\right\|_s$$

$$\frac{\|\delta x\|}{\|x + \delta x\|} \leq \mathcal{K}(A)\frac{\|\delta A\|}{\|A\|}$$

Annexe

Explications détaillées : **conditionnement**

Table 1: Condition numbers before and after equilibration for ALG3 and MC64

matrix	n	nnz of A	$\text{cond}_{orig}(A)$	$\text{cond}_{MC64}(A)$	$\text{cond}_{ALG3}(A)$
dw4096	8192	41746	$1.50 \cdot 10^7$	$9.63 \cdot 10^5$	$5.90 \cdot 10^5$
rajat13	7598	48762	$1.46 \cdot 10^{11}$	$1.62 \cdot 10^1$	$6.07 \cdot 10^2$
utm5940	5940	83842	$1.91 \cdot 10^9$	$2.75 \cdot 10^9$	$3.90 \cdot 10^9$
tols2000	2000	5184	$6.92 \cdot 10^6$	$1.08 \cdot 10^2$	$1.11 \cdot 10^2$
rajat19	1157	3699	$9.17 \cdot 10^{10}$	$5.87 \cdot 10^{11}$	$7.33 \cdot 10^8$
unsym-rand05	1024	1048576	$2.73 \cdot 10^{13}$	$1.77 \cdot 10^5$	$2.27 \cdot 10^5$
unsym-rand04	512	262144	$2.07 \cdot 10^{11}$	$1.85 \cdot 10^5$	$5.66 \cdot 10^5$
unsym-rand03	256	65536	$1.26 \cdot 10^{11}$	$1.80 \cdot 10^4$	$2.78 \cdot 10^4$
unsym-rand02	128	16384	$1.03 \cdot 10^9$	$1.18 \cdot 10^4$	$1.26 \cdot 10^4$
unsym-rand01	64	4096	$1.59 \cdot 10^7$	$1.40 \cdot 10^3$	$2.10 \cdot 10^3$

An exploration of matrix equilibration

Paul Liu, Stanford

Annexe

Explications détaillées : **RK4**

$$\forall t \in \mathbb{R},\ g(f,t) = \frac{df}{dt}(t) \qquad f_0 = f(t_0) \qquad t_n = t_0 + n\tau$$

$$f(t_{n+1}) = f_n = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

$$k_1 = g(f_n,\ t_n)$$

$$k_2 = g(f_n + \frac{\tau}{2}k_1,\ t_n + \frac{\tau}{2})$$

$$k_3 = g(f_n + \frac{\tau}{2}k_2,\ t_n + \frac{\tau}{2})$$

$$k_4 = g(f_n + \tau k_3,\ t_n + \tau)$$