Elastodynamics: Implementation

We are now ready to present pseudocode for implementation of an isogeometric elastodynamics code. For brevity, we present only the HHT-α method. We proceed forward in nearly the same fashion as the linear elasticity setting. The only difference concerns the application of boundary conditions.

For Neumann (i.e., traction) boundary conditions, we utilize the Neumann array to identify element boundaries as before.

For Dirichlet (i.e., displacement) boundary conditions, we utilize the BC array to identify control variables subject to strong boundary conditions. Then, we set:

$$\left.\begin{aligned} M_{PQ} &= \delta_{PQ} \\ F_P(t) &= (\vec{g}_i)_{A,tt} \end{aligned}\right\} \text{ if } BC(A,i) = 1$$

where $P = ID(A,i)$. This corresponds to the acceleration Dirichlet condition:

$$(\vec{d}_i)_{A,tt} = (\vec{g}_i)_{A,tt}$$

Provided initial conditions are provided for $(\vec{d}_i)_A$ and $(\vec{d}_i)_{A,t}$, the above equation is equivalent to $(\vec{d}_i)_A = (\vec{g}_i)_A$. However, the above equation is a second-order ODE in time, and hence we can discretize it in time using the HHT-α method. Thus, we have an equivalent treatment for Dirichlet and non-Dirichlet degrees-of-freedom.

With the above considerations, we can present pseudocode for our implementation. This follows on the next page.

Step 1: Initialize the Problem

Set $p_1, p_2, n_1, n_2, \underline{\underline{\mathcal{M}}}_1, \underline{\underline{\mathcal{M}}}_2, \{\vec{P}_i\}_{i=1}^{n}, \{w_i\}_{i=1}^{n}$

Call Extract And Localize to obtain $n_{el}, \{\underline{\underline{C}}^e\}_{e=1}^{nel}, IEN, \{\{\vec{P}_a^{be}\}\}, \{\{w_a^{be}\}\}$

Set $n_q, \{\hat{\xi}_q\}_{q=1}^{n_q}, \{\hat{w}_q\}_{q=1}^{n_q}$

Construct ID and LM arrays

Construct BC and Neumann arrays

Construct $g_A, \dot{g}_A, \ddot{g}_A$, $A = 1, \dots, d$ (time-dependent vectors w/ Dirichlet data)

Set the number of time steps n_{step} and time instances $\{t_n\}_{n=0}^{nstep}$

Set $\gamma, \beta, \alpha, \varepsilon$ (parameters for HHT-α)

Step 2: Set the Initial Conditions

Solve for the initial conditions $\vec{u}_0^h$ and $\vec{v}_0^h$ and thus set $\underline{d}_0$ and $\underline{v}_0$

Set $\underline{a}_0$ by, for example, solving the system $\underline{\underline{M}}\,\underline{a}_0 + \underline{\underline{C}}\,\underline{v}_0 + \underline{\underline{K}}\,\underline{d}_0 = \underline{F}_0$

Step 3: Evolve the System in Time

for $n = 0, \dots, n_{step} - 1$

Form and Assemble $\underline{\underline{M}}, \underline{\underline{C}}, \underline{\underline{K}}$, & $\underline{F}_{n+1}$ **

Predictor Step: Set $i = 0$
Set $\underline{d}_{n+1}^{i} = \hat{\underline{d}}_{n+1}$
Set $\underline{v}_{n+1}^{i} = \hat{\underline{v}}_{n+1}$
Set $\underline{a}_{n+1}^{i} = \hat{\underline{a}}_{n+1}$

Set flag = 0

while flag == 0

Set Intermediate Values: Set $\underline{d}_{n+\alpha}^{i} = \underline{d}_n + \alpha(\underline{d}_{n+1}^{i} - \underline{d}_n)$
Set $\underline{v}_{n+\alpha}^{i} = \underline{v}_n + \alpha(\underline{v}_{n+1}^{i} - \underline{v}_n)$

Set Residual: Set $\Delta\underline{F}_{n+1}^{i} = \underline{F}_{n+1} - \underline{\underline{M}}\,\underline{a}_{n+1}^{i} - \underline{\underline{C}}\,\underline{v}_{n+\alpha}^{i} - \underline{\underline{K}}\,\underline{d}_{n+\alpha}^{i}$

Set Update: Solve $\underline{\underline{M}}^* = \underline{\underline{M}} + \gamma\Delta t\,\underline{\underline{C}} + \beta\Delta t^2\,\underline{\underline{K}}$ $\underline{\underline{M}}^*\,\Delta\underline{a} = \Delta\underline{F}_{n+1}^{i}$

Corrector Step: Set $\underline{a}_{n+1}^{i+1} = \underline{a}_{n+1}^{i} + \Delta\underline{a}$
Set $\underline{v}_{n+1}^{i+1} = \underline{v}_{n+1}^{i} + \gamma\Delta t\,\Delta\underline{a}$
Set $\underline{d}_{n+1}^{i+1} = \underline{d}_{n+1}^{i} + \beta\Delta t^2\,\Delta\underline{a}$

```
        if ||ΔF^i_{n+1}|| ≤ ε ||ΔF^0_{n+1}||
            Set flag=1
        else
            Set i=i+1
        endif

    endwhile

    Set d_{n+1} = d^{i+1}_{n+1}
    Set v_{n+1} = v^{i+1}_{n+1}
    Set a_{n+1} = a^{i+1}_{n+1}

endloop
```

Step 4: Analyze Results

In the above pseudocode, we form $\underline{\underline{M}}$, $\underline{\underline{C}}$, $\underline{\underline{K}}$, & $\underline{F}_{n+1}$ at each time step. This procedure looks as follows:

```
Form and Assemble:

    Set M = 0, C = 0, K = 0, and F_{n+1} = 0

    for e=1,...,nel

        Call Element Formation to obtain m^e, c^e, k^e, and f^e_{n+1}

        Call Element Assembly to obtain updated M, C, K, and F_{n+1}

    endloop

    for i=1,...,n
        for A=1,...,d
            if BC(A,i) = 1
                Set P = ID(A,i)
                Set M_pp = 1
                Set (F_{n+1})_P = (q_i)_{A,tt}
            endif
        endloop
    endloop
```

The functions Element Formation and Element Assembly are detailed on the following pages.

Element Assembly:

for $A = 1, \ldots, d$
  for $a = 1, \ldots, n_{loc}$
    Set $p = d(a-1) + A$
    Set $P = LM(A, a, e)$
    Set $i = IEN(a, e)$

    if $BC(A, i) = 0$
      for $B = 1, \ldots, d$
        for $b = 1, \ldots, n_{loc}$
          Set $q = d(b-1) + B$
          Set $Q = LM(B, b, e)$
          Set $j = IEN(b, e)$

          if $BC(B, j) = 0$
            Update $M_{PQ} = M_{PQ} + m^e_{pq}$

            Update $C_{PQ} = C_{PQ} + c^e_{pq}$

            Update $K_{PQ} = K_{PQ} + k^e_{pq}$

          else
            Update $(\underline{F}_{n+1})_p = (\underline{F}_{n+1})_p$
              $- m^e_{pq} (\vec{g}_j)_{B,tt}$
              $- c^e_{pq} (\vec{g}_j)_{B,t}$
              $- k^e_{pq} (\vec{g}_j)_B$

          endif
        endloop
      endloop

      Update $(\underline{F}_{n+1})_p = (\underline{F}_{n+1})_p + (\underline{f}^e_{n+1})_p$
    endif
  endloop
endloop

Element Formation:

Initialize $\underline{\underline{m}}^e = \underline{\underline{0}}$, $\underline{\underline{c}}^e = \underline{\underline{0}}$, $\underline{\underline{k}}^e = \underline{\underline{0}}$, $\underline{f}^e_{n+1} = \underline{0}$

for $q_1 = 1, \ldots, n_q$
  for $q_2 = 1, \ldots, n_q$

Call Shape Function $(e, \hat{\xi}_{q_1}, \hat{\xi}_{q_2})$ to obtain R^e_a, $\frac{\partial}{\partial x} R^e_a$, $\frac{\partial}{\partial y} R^e_a$, $\vec{x}$, and $\underline{\underline{J}}$ at $\vec{x}^e(\hat{\xi}_{q_1}, \hat{\xi}_{q_2})$

Set $j \equiv \det(\underline{\underline{J}})$
Set $\vec{\vec{D}}_{loc} = \vec{\vec{D}}(\vec{x}, t)$
Set $\rho_{loc} = \rho(\vec{x}, t)$
Set $\vec{f}_{loc} = \vec{f}(\vec{x}, t)$

for $a = 1, \ldots, n_{loc}$

Form $\vec{\vec{B}}^e_a = \begin{bmatrix} N^e_{a,1} & 0 \\ 0 & N^e_{a,2} \\ N^e_{a,2} & N^e_{a,1} \end{bmatrix}$

for $b = 1, \ldots, n_{loc}$

Form $\vec{\vec{B}}^e_b = \begin{bmatrix} N^e_{b,1} & 0 \\ 0 & N^e_{b,2} \\ N^e_{b,2} & N^e_{b,1} \end{bmatrix}$

Set $\underline{\underline{k}}_{loc} = (\vec{\vec{B}}^e_a)^T \vec{\vec{D}}_{loc} \vec{\vec{B}}^e_b * \hat{w}_{q_1} * \hat{w}_{q_2} * j$
Set $m_{loc} = N^e_a * \rho_{loc} * N^e_b * \hat{w}_{q_1} * \hat{w}_{q_2} * j$
for $A = 1, \ldots, d$
  for $B = 1, \ldots, d$

Set $p = d(a-1) + A$
Set $q = d(b-1) + B$

Update $m^e_{pq} = m^e_{pq} + \delta_{AB} * m_{loc}$

Update $k^e_{pq} = k^e_{pq} + \hat{e}^T_A \underline{\underline{k}}_{loc} \hat{e}_B$

  endloop
endloop

endloop

for $A = 1, \ldots, d$

Set $p = d(a-1) + A$
Update $(\underline{f}^e_{n+1})_p = (\underline{f}^e_{n+1})_p + R^e_a * (\vec{f}_{loc})_A * \hat{w}_{q_1} * \hat{w}_{q_2} * j$

endloop

endloop

endloop

endloop

for $A = 1, \ldots, d$

for side $= 1, \ldots, 4$

if Neumann $(A, \text{side}, e) = 1$

for $q = 1, \ldots, n_q$

Set $\tilde{\xi}_1 = \begin{cases} \tilde{\xi}_q & \text{if side} = 1,3 \\ 1 & \text{if side} = 2 \\ 0 & \text{if side} = 4 \end{cases}$

Set $\tilde{\xi}_2 = \begin{cases} \tilde{\xi}_q & \text{if side} = 2,4 \\ 1 & \text{if side} = 3 \\ 0 & \text{if side} = 1 \end{cases}$

Set $\underline{\tilde{t}} = \begin{cases} \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \text{if side} = 1,3 \\ \begin{bmatrix} 0 \\ 1 \end{bmatrix} & \text{if side} = 2,4 \end{cases}$

Call Shape Function $(e, \tilde{\xi}_1, \tilde{\xi}_2)$ to obtain $R_a^e, \frac{\partial}{\partial x} R_a^e, \frac{\partial}{\partial y} R_a^e, \vec{x}, \underline{\underline{J}}$ at $\vec{x}^e(\tilde{\xi}_1, \tilde{\xi}_2)$

Set $j_{\Gamma^2} = |\underline{\underline{J}} \underline{\tilde{t}}|$

Set $h_{A\text{loc}} = h_A(\vec{x}, t)$

for $a = 1, \ldots, n_{\text{loc}}$

Set $p = d(a-1) + A$

Update $(f_{n+1}^e)_p = (f_{n+1}^e) + R_a^e * h_{A\text{loc}} * \tilde{w}_q * j_{\Gamma^2}$

endloop

endloop

endif

endloop

endloop

Set $\underline{\underline{C}}^e$ if Rayleigh damping is applied