June 13, 2021

Algorithms

Mattias Fredriksson

1 Algorithms

To determine a set of algorithms for solving the inverse kinematics problem, solutions can be tailored to solve specific problems. Depending on the possible system and end effector configurations, three scenarios are covered:

- An open serial link chain with end effectors parameterized by position coordinates [*section.*1.1].
- A serial link chain on the form of a tree structure (forming multiple open chains), controlled by end effectors position coordinates [*section.*1.2].
- An open serial link chain consisting of spheroid joints controlled by a single end effector (each spheroid joint is constructed from 3 revolute joints). End effectors are parameterized by independent position and exponential coordinates [*section.*1.3].

Each approach is discussed in a separate section as referenced above. Algorithms are also constructed in consideration to a row-major memory alignment (similar to the C language) and designed for simple Numpy (Python) implementation. Before going into the IK algorithms themselves, an example of the Gauss-Newton optimization algorithm is presented in *algo.*1, and some of the notation used can be summarized by:

- A_i - Either represent access of the i:th element in a list/tensor or i:th column of a matrix. First element is indexed by 0 to match general code conventions.
- $A_{i,:}$ - Access of the i:th row of a matrix (unintuitive with regard to a row-major memory alignment but is used to match the math notation).
- $0_{n\times m}$ - Matrix of zeros with n columns and m rows.
- $\mathbb{1}_{n\times m}$ - Matrix of ones with n columns and m rows.
- $\{\{a, b\}, \dots\}$ - A list of ordered sets, with each subset containing elements a and b.
- $A \cup B$ - Insertion of B as the last element in the list/set A.
- $algo_i(\dots)$ - Function call referring to the i:th algorithm using some input arguments.
- # - Comment, used to highlight implementation details but are sparsely used.

Algorithm 1: A Gauss-Newton optimization algorithm.

Input : System function JE returning the Jacobian and residual, initial condition $\boldsymbol{q}_0$, maximum iteration count m, residual threshold ϵ, and system arguments $args$.

Output: Optimized $\boldsymbol{q}$ and residual $\boldsymbol{r}$.

begin
  while $i < m$ **do**
    $\{ J, \boldsymbol{r} \} \leftarrow JE(\boldsymbol{q}, args)$
    if $\|\boldsymbol{r}\| < \epsilon$ **then**
      break
    end
    $\boldsymbol{\delta} \leftarrow J^{-1} y$ `// Use an appropriate solver when` J^{-1} `is not square.`
    $\boldsymbol{q} \leftarrow \boldsymbol{q} - \boldsymbol{\delta}$ `// Subtract` $\boldsymbol{\delta}$ `since` y `should be negated.`
  end
end

1.1 Open revolute link chain

Algorithm for a simple system consisting of a set of revolute joints. End effectors are defined by position coordinates sampled in task space and modeled as translative offsets in joint space. Parameters used within this section's algorithms are:

- $\boldsymbol{q}$: Configuration state for the system as defined by the generalized coordinates. For this section, the vector only contains angles associated with revolute joints.
- B: Set of rigid body parameters on form $\{\{\hat{\boldsymbol{k}},\ \boldsymbol{p}_c\},\ \ldots\}$, containing rotation axis $\hat{\boldsymbol{k}}$ and translation offset $\boldsymbol{p}_c$.
- O: Set of end effector parameters on form $\{\{j, \boldsymbol{p}_e\},\ \ldots\}$, with elements containing a translation offset $\boldsymbol{p}_e$ relative to the j:th joint.
- S: Intended end effector positions $\boldsymbol{p}_s$ sampled in task space where $S = \left|\boldsymbol{p}_{s_1},\ \boldsymbol{p}_{s_2},\ \ldots,\ \boldsymbol{p}_{s_n}\right|^T$.

Algorithm 2: Compute the set of joint parameters W in task space.

Input : Configuration $\boldsymbol{q}$ and chain parameters B.
Output: Ordered set of joint parameters W.
begin
  $R \leftarrow I$
  $\boldsymbol{c} \leftarrow 0_{3\times1}$
  $W \leftarrow \{\ \}$
  for $\{\hat{\boldsymbol{k}}, \boldsymbol{p}\} \in B$ **and** $a \in \boldsymbol{q}$ **do**
    $c \leftarrow R\boldsymbol{c} + \boldsymbol{p}$
    $R \leftarrow Re^{[\hat{\boldsymbol{k}}]_\times a}$
    $W \leftarrow W \cup \{R\hat{\boldsymbol{k}}, R, \boldsymbol{c}\}$
  end
end

Algorithm 3: Compute position of modeled end effectors in task space.

Input : Ordered set of joint parameters W and end effector parameters O.
Output: Matrix $E \in \mathbb{R}^{n\times3}$, containing n rows of end effector positions in task space.
begin
  $E \leftarrow 0_{n\times3}$
  $i \leftarrow 0$
  for $\{j, \boldsymbol{p}_e\} \in O$ **do**
    $\boldsymbol{\omega}, R, \boldsymbol{c} \leftarrow W_j$
    $E_{i,:} \leftarrow R\boldsymbol{p}_e + \boldsymbol{c}$
    $i \leftarrow i + 1$
  end
end

Algorithm 4: Compute index matrix I.

Input : Set of end effector parameters O.
Output: Matrix $I \in \mathbb{R}^{n \times 1}$, containing joint index associated with the n end effectors.
begin
  $i \leftarrow 0$
  for $\{ j, \boldsymbol{p}_e \} \in O$ **do**
    $I_i \leftarrow j$
    $i \leftarrow i + 1$
  end
end

Algorithm 5: Compute Jacobian.

Input : Ordered set of joint parameters W, end effector positions E, and index matrix I.
Output: Jacobian $J \in \mathbb{R}^{3n \times m}$.
begin
  $J \leftarrow 0_{3n \times m}$
  for $\{ \boldsymbol{\omega}, R, \boldsymbol{c} \} \in W$ **do**
    $X \leftarrow (E - \mathbb{1}_{m \times 1} \boldsymbol{c}^T)[-\omega]_\times$
    $X \leftarrow X \circ (j <= I)\mathbb{1}_{1 \times 3}$ # Elementwise less equal.
    $J_i \leftarrow \left| X_{0,:} \quad X_{1,:} \quad \dots \quad X_{n-1,:} \right|^T$
    $j \leftarrow j + 1$
  end
end

Algorithm 6: Compute system Jacobian and residual.

Input : System configuration $\boldsymbol{q}$, rigid body parameters B, end effector parameters O, and intended end effector configuration S.
Output: Jacobian $J \in \mathbb{R}^{3n \times m}$, and residual vector $\boldsymbol{r} \in \mathbb{R}^{3n}$.
begin
  $W \leftarrow algo_2(\boldsymbol{q}, B)$
  $E \leftarrow algo_3(W, O)$
  $I \leftarrow algo_4(O)$
  $J \leftarrow algo_5(W, E, I)$
  $D \leftarrow E - S$
  $\boldsymbol{r} \leftarrow \left| D_{0,:} \quad D_{1,:} \quad \dots \quad D_{n-1,:} \right|^T$
end

Optimization of a configuration state using the Gauss-Newton optimizer in *algo.*1 is then possible by feeding the optimizer appropriate function *algo.*6 for computing the Jacobian and residual for the current configuration, using appropriate system parameters.

1.2 Open link chain hierarchy

Algorithms for a system consisting of a hierarchy of revolute and prismatic joints with end effector defined by position coordinates. Note that the algorithms are constructed as an extended version from previous section. Parameters used within this section's algorithms are:

- $\boldsymbol{q}$: Configuration state for the system as defined by the generalized coordinates. In this section parameters represent either a translation along direction $\hat{\boldsymbol{d}}$ or rotation around axis $\hat{\boldsymbol{k}}$, or both (for screw joints).
- B: Set of rigid body parameters on form $\{\{\hat{\boldsymbol{k}}, \hat{\boldsymbol{d}}, R_c, \boldsymbol{p}_c\}, \ldots\}$, containing parent link i, rotation axis $\hat{\boldsymbol{k}}$, translation direction $\hat{\boldsymbol{d}}$ and transformation offsets R_c and $\boldsymbol{p}_c$.
- O: Set of end effector parameters on form $\{\{j, \boldsymbol{p}_e\}, \ldots\}$, with elements containing a translation offset $\boldsymbol{p}_e$ relative to the j:th joint.
- S: Intended end effector positions $\boldsymbol{p}_s$ sampled in task space where $S = \left|\boldsymbol{p}_{s_1}, \boldsymbol{p}_{s_2}, \ldots, \boldsymbol{p}_{s_n}\right|^T$.

Algorithm 7: Compute the set of joint parameters W in task space.

Input : Configuration $\boldsymbol{q}$ and rigid body parameters B.
Output: Ordered set of joint parameters W.
begin
  $R \leftarrow 0_{(m+1)\times 3\times 3}$
  $C \leftarrow 0_{(m+1)\times 3\times 1}$
  $W \leftarrow \{\}$
  $R_0 \leftarrow I$
  $j \leftarrow 1$
  for $\{i, \hat{\boldsymbol{k}}, \hat{\boldsymbol{d}}, R_c, \boldsymbol{p}_c\} \in B$ **and** $q_j \in \boldsymbol{q}$ **do**
    $R_c \leftarrow R_{i+1} R_c$
    $C_j \leftarrow R_{i+1}\boldsymbol{p}_c + C_{i+1} + q_j R_c \hat{\boldsymbol{d}}$
    $R_j \leftarrow R_c e^{[\hat{\boldsymbol{k}}]_\times q_j}$
    $W \leftarrow W \cup \{R\hat{\boldsymbol{k}}, R_c\hat{\boldsymbol{d}}, R_j, C_j\}$
    $j \leftarrow j + 1$
  end
end

Algorithm 8: Compute position of modeled end effectors in task space.

Input : Ordered set of joint parameters W and end effector parameters O.
Output: Matrix $E \in \mathbb{R}^{n\times 3}$, containing n rows of end effector positions in task space.
begin
  $E \leftarrow 0_{n\times 3}$
  $i \leftarrow 0$
  for $\{j, \boldsymbol{p}_e\} \in O$ **do**
    $\boldsymbol{\omega}, \hat{\boldsymbol{d}}, R, \boldsymbol{c} \leftarrow W_j$
    $E_{i,:} \leftarrow R\boldsymbol{p}_e + \boldsymbol{c}$
    $i \leftarrow i + 1$
  end
end

Algorithm 9: Compute dependency matrix I.

Input : Rigid body parameters B and end effector parameters O.
Output: Dependency matrix $I \in \mathbb{R}^{m \times n}$, i:th row entry is marked by 1 if the i:th end effector is dependent joint associated with the row (0 represents independence).

begin
$X \leftarrow 0_{m,m}$
$I \leftarrow 0_{m,n}$
$j \leftarrow m - 1$
# Form dependency matrix X for the joint hierarchy.
while $j >= 1$ **do**
$\{ i, \hat{\boldsymbol{k}}, \hat{\boldsymbol{d}}, R_c, \boldsymbol{p}_c \} \leftarrow B_j$
$X_{j,j} = 1$
$X_{i,:} \leftarrow X_{i,:} \parallel X_{j,:}$ # Elementwise/bitwise or.
$j \leftarrow j - 1$
end
$X_{0,0} = 1$

# Form end effector dependency matrix.
for $\{ j, \boldsymbol{p}_e \} \in O$ **do**
$I_i \leftarrow X_j$ # Dependencies are defined in columns of X.
end
end

Algorithm 10: Compute Jacobian.

Input : Ordered set of joint parameters W, end effector positions E, and dependency matrix I.
Output: Jacobian $J \in \mathbb{R}^{3n \times m}$.

begin
$J \leftarrow 0_{3n \times m}$
$j \leftarrow 0$
for $\{ \boldsymbol{\omega}, \hat{\boldsymbol{d}}, R, \boldsymbol{c} \} \in W$ **do**
$X \leftarrow (E - \mathbb{1}_{m \times 1} \boldsymbol{c}^T)[-\omega]_\times$
$X \leftarrow I_{j,:} \mathbb{1}_{1 \times 3} \circ X$
$J_i \leftarrow \begin{vmatrix} X_{0,:} & X_{1,:} & \dots & X_{n-1,:} \end{vmatrix}^T$
$j \leftarrow j + 1$
end
end

Algorithm 11: Compute system Jacobian and residual.

Input : System configuration $\boldsymbol{q}$, rigid body parameters B, end effector parameters O, and intended end effector configuration S.

Output: Jacobian $J \in \mathbb{R}^{3n \times m}$, and residual vector $\boldsymbol{r} \in \mathbb{R}^{3n}$.

begin

$W \leftarrow algo_7(\boldsymbol{q}, B)$

$E \leftarrow algo_8(W, O)$

$I \leftarrow algo_9(B, O)$

$J \leftarrow algo_{10}(W, E, I)$

$D \leftarrow E - S$

$\boldsymbol{r} \leftarrow \begin{vmatrix} D_{0,:} & D_{1,:} & \dots & D_{n-1,:} \end{vmatrix}^T$

end

1.3 Open spheroid link chain with target in $\mathbb{R}^6$

Algorithm for a system consisting of an open link chain of spheroid (ball and socket) joints with an end effector defined by both position and orientation coordinates. Algorithm for computing the Jacobian and residual is split in two parts which can be combined. Parameters used in this section are:

- $\boldsymbol{q}$: Configuration state for the system as defined by the generalized coordinates. System is parameterized using Euler angles.
- B: Set of rigid body parameters on form $\{\{R_c, \boldsymbol{p}_c\}, \ldots\}$, with joint sets only determined through transformation offsets R_c and $\boldsymbol{p}_c$.
- O: The end effector is parameterized through the pair $\{\boldsymbol{p}_e, \boldsymbol{\omega}_e\}$, defining displacement from the last joint (using both a translation and orientation).
- S: Intended end effector configuration is parameterized in fixed space through the coordinate pair $\{\boldsymbol{p}_{sT}, \boldsymbol{\omega}_{sT}\}$.

Algorithm 12: Compute system configuration parameters.

Input : System configuration $\boldsymbol{q}$, rigid body parameters B.
Output: Joint positions $P \in \mathbb{R}^{3\times m}$, joint rotation axis $W \in \mathbb{R}^{3\times m}$, rotation of final joint R and its position $\boldsymbol{c}$.

begin
  $P \leftarrow 0_{3\times m}$
  $W \leftarrow 0_{3\times m}$
  $R \leftarrow I$
  $\boldsymbol{c} \leftarrow \boldsymbol{0}$

  # Compute configuration in task space.
  $j \leftarrow 0$
  for $\{R_c, \boldsymbol{p}_c\} \in B$ **do**
    $\boldsymbol{c} = R_c\boldsymbol{p}_c + \boldsymbol{c}$
    $R \leftarrow RR_c$
    for $i \in 1, 2, 3$ **do**
      $R \leftarrow Re^{[\boldsymbol{e}_i]_\times \boldsymbol{q}_j}$
      $W_j \leftarrow R_i$
      $P_j \leftarrow \boldsymbol{c}$
      $j \leftarrow j + 1$
    end
  end
end

Algorithm 13: Compute system Jacobian and residual.

Input : System configuration $\boldsymbol{q}$, rigid body parameters B, end effector parameters O, and intended end effector configuration S.

Output: Jacobian $J \in \mathbb{R}^{3n \times m}$, and residual vector $\boldsymbol{r} \in \mathbb{R}^{3n}$.

begin

$\{P,\ W,\ R,\ \boldsymbol{c}\} \leftarrow algo_{12}$

$\{\boldsymbol{p}_e,\ \boldsymbol{\omega}_e\} \leftarrow O$

$\{\boldsymbol{p}_{sT},\ \boldsymbol{\omega}_{sT}\} \leftarrow S$

Compute residual $\boldsymbol{i}_s(\boldsymbol{q}) - \boldsymbol{i}_s(\boldsymbol{q}_T)$.

$\boldsymbol{p}_{se} \leftarrow R\boldsymbol{p}_e + \boldsymbol{c}$

$\boldsymbol{\omega}_{\boldsymbol{r}} \leftarrow log(e^{[\omega_{sT}]_\times} e^{[-\omega_e]_\times} R^T)$

$\boldsymbol{r} \leftarrow \begin{vmatrix} \boldsymbol{p}_{se} - \boldsymbol{p}_{sT} \\ -\boldsymbol{\omega}_{\boldsymbol{r}} \end{vmatrix}$

Compute Jacobian.

$J_{\boldsymbol{p}} \leftarrow \begin{vmatrix} [W_0]_\times(\boldsymbol{p}_{se} - P_0) & [W_1]_\times(\boldsymbol{p}_{se} - P_1) & \dots & [W_{m-1}]_\times(\boldsymbol{p}_{se} - P_{m-1}) \end{vmatrix}$

$J \leftarrow \begin{vmatrix} J_{\boldsymbol{p}} \\ W \end{vmatrix}$

end