Thesis overview

Ashesh Vasalya

April 1, 2019

Algorithm 1: linear prediction controller

Input: $\mathcal{T}_R^{efL}, mocapData$

Output: wp_{efL}^{obj} `// predicted waypoints`

Data: Initial require: $t_{observe} = 150ms, t_{predict} = 3sec, i = 1$

i++ `// increments as per controller run-time (5ms)`

if $(i\%t_{observe}) == 0$ **then**

  $\mathcal{T}_R^{efL} \leftarrow endEffectorTask$

  $\mathcal{T}_M^{efL_{marker}} \leftarrow mocapData.robotMarker((i - t_{observe}) + 1)$

for $j \leftarrow 1$ *to* $t_{observe}$ **do**

  $\mathcal{T}_M^{obj_{marker}} \leftarrow mocapData.objectMarker(i - t_{observe}) + j)$

  $\mathcal{T}_{efL}^{obj_{marker}} = {\mathcal{T}_R^{efL}}^{-1} \times \mathcal{T}_M^{obj_{marker}} \times {\mathcal{T}_M^{efL_{marker}}}^{-1} \times \mathcal{T}_R^{efL}$

  if $j == t_{observe}$ **then**

    $P_{t_{observe}}^{obj_{marker}} \leftarrow \mathcal{T}_{efL}^{obj_{marker}}.translation()(t_{observe})$

$\bar{\mathcal{V}}_{obj_{marker}}^{efL} \leftarrow \mathcal{F}_{avg}(\mathcal{T}_{efL}^{obj_{marker}}.translation())$

`/* predict position of object handover at` $t_{predict}$ `*/`

$P_{t_{predict}}^{obj} \leftarrow \bar{\mathcal{V}}_{obj_{marker}}^{efL} \times t_{predict} + P_{t_{observe}}^{obj_{marker}}(t_{observe})$

`/* generate way points between robot left end effector and object handover location */`

Function `generateWp`$(P_{t_{observe}}^{obj_{marker}}, P_{t_{predict}}^{obj}, t_{predict})$**:**

  for $k \leftarrow 0$ *to* $t_{predict}$ **do**

    $wp_{efL}^{obj}(k) \leftarrow (P_{t_{predict}}^{obj} - P_{t_{observe}}^{obj_{marker}}) \times k + P_{t_{observe}}^{obj_{marker}}$

  return wp_{efL}^{obj}

Our prediction controller behavior can be tuned by two initially required constant time periods, $t_{observe}$ —which defines the time period required to observe the motion of object and $t_{predict}$ —required to predict the object handover location in advance.

Inputs of the controller are robot left end effector pose $\mathcal{T}_R^{efL}$ and mocap markers position data in the mocap frame of reference $mocapFrameData$. $\mathcal{T}_M^{efL_{marker}}$ is the left end effector marker pose in the mocap frame. Similarly, object marker pose given by $\mathcal{T}_M^{obj_{marker}}$. For simplicity, at the moment, we have assumed zero rotation of the markers and end effector, therefore rotation part of $\mathcal{T}_R^{efL}$, $\mathcal{T}_M^{efl_{marker}}$ and $\mathcal{T}_M^{obj_{marker}}$ are Identity matrix $\mathcal{I}$.

The transformation matrix $\mathcal{T}_{obj_{marker}}^{efL}$, provides the relative pose of object marker w.r.t. robot left end effector in the robot coordinate system. $P_0^{obj_{marker}}$ $P_{t_{observe}}^{obj_{marker}}$ are the updated observed positions of object whenever the condition $(i\%t_{observe} == 0)$ satisfy. Based on the $P_0^{obj_{marker}}$ $P_{t_{observe}}^{obj_{marker}}$, average velocity $\bar{\mathcal{V}}_{obj_{marker}}^{efL}$ of the observed object motion calculated. Where, $\mathcal{F}_c$, $\mathcal{F}_{diff}$, $\mathcal{F}_{avg}$ are the *helper functions*.

Function $PREDICTPOS$, returns the predicted position of the handover $P_{t_{predict}}^{obj}$ at time $t_{predict}$.

Function $GENERATEWP$, returns the way-points wp_{obj}^{efL} between robot left end effector and object handover location, which is the final output of the controller.

Later, wp_{obj}^{efL} is being fed in the `mc_rtc positionTask`.

Algorithm 2: Force Based Gripper Controller

Input: $\mathcal{F}_w$ `// EF wrist worldWrenchWithoutGravity`

Output: $\mathcal{F}_{pull}, T_{new}$ `// Pull force, new threshold based on object mass`

i++ `// increments as per controller run-time (5ms)`

if *subject hand is near robot* **then**

  `/* when SUBJECT holds the object */`

  if $\mathcal{F}_w.norm() < 1.0$ `// gripper is empty`

  then

    $\mathcal{F}_{zero} \leftarrow \mathcal{F}_w$ `// wrench offset`

  else if $\mathcal{F}_w.norm() > 2.0$ **then**

    $\mathcal{F}_{object} \leftarrow \mathcal{F}_w$ `// wrench with object`

  `/* when ROBOT holds the object */`

  Function `CheckPullForce(`$axis \forall x,y,z$`):`

    $objectMass \leftarrow \mathcal{F}_{load}/9.81$ `// get object mass`

    $\mathcal{F}_{inert} \leftarrow objectMass * efAce$ // $efAce$ `using gripper markers`

    $\mathcal{F}_{pull} \leftarrow |(|F_w - |F_{inert})$

    $T_{new} \leftarrow \mathcal{F}_{load} + T_{old}$ // T_{old} `set to min by user`

    if $\mathcal{F}_{pull} > T_{new}\ \forall x,y,z$ **then**

      openGripper `// release object`

else

  if $(i\%200)$ **then**

    $\mathcal{F}_{load} \leftarrow |(\mathcal{F}_w - \mathcal{F}_{zero})$ `// ∀x,y,z average over time`