Before actuate the prototype:

Sensor Calibration 🡪 Sensor Initial Value Calibration 🡪 Sensor Feedback Process and Uploaded Info Package Checking 🡪 Power Supply Checking 🡪 MCU Electrical Cable Connection Checking 🡪 Motor Operation Direction Checking 🡪 The Following Checking Terms Mentioned in Test Conclusion.docx and Test Reminder.docx

20210408

* The serial buffer is defined in core head file '\\....\Arduino\hardware\Arduino\_STM32\STM32F1\system\libmaple\include\libmaple\usart.h'.
* There may occurs data overflow in MCU receive buffer if high-level control frequency does not match the low-level control frequency as no buffer clear program is written for MCU

20210409

* When high-level controller is stopped with timer object delete in PC high-level controller, a stop command will be sent to MCU for low-level controller, but no guarantee measurement has be made that MCU can correctly receive this command
* Potential measurement-Make sure the main program only stop when info package feedback turn to resume to a safe operation point

20210410

* Caution that before applying the control program, each sensor should be calibrated first
* IMU, load cell, spring stiffness, motor actuation unit parameter, system parameter
* IMU operation algorithm: 6 axis/9 axis
* To adjust sensor feedback item for control, both ADC config and sensor feedback info prcessing program should be adjusted simultaneously to make sure it is obtained correctly with correct command
* ADC enable channel
* ADC detected channel
* IMU operation algorithm (6/9axis)
* Sensor feedback direction (+/- sign is coincides with controller definition or not)
* Sensor feedback processing items in sensorFeedbackPro()
* To increase main-loop running speed, Enabled ADC channel can be adjusted.
* Adjust of ENABLED\_CH & *i* in getADC() and
* Adjust of ENABLED\_CH & *i* & *t* in getADCaverage()
* Before running the program, the motor actuation parameter and rotation direction need to be calibrated
* Motor rotation direction
* Actuation unit parameters:
* Motor current constant, motor driver configuration, gear box
* Pulley radius
* Spring stiffness
* For the Motor enable/disable logic: the motor enable pin is disabled after initialization in MCU program 🡪 will be enabled when the 'mode' flag received from PC indicates normal operation 🡪 will be disable again if ‘mode’ flag indicates stop status
* The stop condition of high-level controller need to be adjusted as practical application required
* For testing: time condition may be enough
* For practical application: At present, the condition can be time condition + mode status condition

20210415

* Yaw angle return to zero logic
* Need to be updated with practical user intention detection strategy
* Timer set in MATLAB Prog
* ‘BusyMode’ : ‘queue’ or ‘drop’
* Frequency adjustment for highest communication frequency
* Serial port set in MATLAB Prog
* Terminator
* BaudRate
* Com port name

202010416

* At present the test bench version program only contains parameters of actuation unit and partial sensors of one side torque transmission system, parameter of another side torque transmission system needs to be added.

20210417

* The moving average/exponential filter need to be adjusted for target ADC feedback processing
* Check if the communication protocol is coincided to communication program set up like: terminator, SendItemFlag set up.

20210521

* The RTG strategy for cable-driven system may follow the process: Actuation system shut off (upright posture) 🡪 Small reference torque for cable tension (slight bending) 🡪 Normal RTG strategy (deep bending)
* Remember to saving data

20210522

* The 'mode' meaning in MCU program should be adjusted along with the UID strategy

20210524

* With handshake logic, no matter power on high-level controller first or low-level controller first

20210610

* The friction compensation term and strategy logic need to be double checked
* During lowering, friction is our friend since the cable intends to extension with lowering motion;
* During lifting with assistive torque increasing, friction is not our friend since the support beam intends to bending more with cable is pulled by motor;
* During lifting with assistive torque decreasing, friction becomes to our friend again as the support beam intends to rotate back while the friction is stopping it rorates back to remaining a large torque
* When friction is our friend during lifting, the invension command of motor need to be constrained to happened while the support beam & torsion spring
* Load Cell shaking more serious when the motor is rotating: noisy isolation related to motor is needed

20210611

* The limitation threshold for control input and delta\_control input should be reasonable
* Due to the friction, PD control with friction compensation have some processing logic: 1) Enable motor to actively reverse when support beam rotation angle is larger than certain value so that the cable can be extended itself with torsion spring's torque; 2) The delta\_Ta = delta\_FrictionCompensation + delta\_PID is limited and can be separatly limited or limited as a whole term.

20210719

* Since during the Stop State: the initial value and control auxiliary including the phase indicator, mode indicator will be initialized, the controller output will be zero or at least near zero. For Exit state: only controller structure and IO are initialized, the sensor works normally, so the feedback of the low-level controller is normal despite the initialization of the desired torque. Therefore the control output might not be zero but the motor is shut off so no operation will be conducted 🡪 20210812 The control output is forced to be zero now

20210920: PID Implementation Related Issues

* The limitation is (can be) set for the following value:
* Total Ta
* Delta Ta
* P components
* Every kinds of compensation term
* The ID gains of PID controller are set under the intrinsic assumption of Tcontrol = 1s instead of real control period for de/dt = (e(k) - e(k-1))/Tcontrol, which makes the practical effect Kd value is 0.01 times of set value
* If there is no delta-limitation of Ta, then this incremental style PID is equal to positional style PID
* If there is a small delta-limitation of Ta, then it may limit the normal increasing of Ta from PID when there is a large tracking error occurs
* An adjust mechanism is added that: if delta\_Ta reached the delta limitation, then a residual term is calculated as (pidR.ResDelta\_Ta = pidR.Delta\_Ta - LimitDelta\_TaR\*Value\_sign(pidR.Delta\_Ta)) and it will store the remaining delta\_Ta for next cycle’s control output updating.
* From previous experiments experience:
* Test bench experiments without fixed HB link, no delta\_Ta limitation is set and the D gain of PID is larger than that for prototype
* Prototype experiments with human bending and impedance Tr, a small delta\_Ta limitation is set with a smaller D gain and without adjust mechanism for this time’s error also works well

20210921

* Basic UID Threshold selection attention
* Standing -> Walking(1) v.s. Standing -> Lowering(2)
* angle(1) < angle(2)
* Standing -> Lowering
* ab\_angle > angle
* Standing -> Lowering (1) v.s. Lowering -> Grasping (2)
* angle(1) < angle(2)
* velocity(1) > velocity(2) > consist\_velocity(2)
* Standing -> Lowering (1) v.s. Lowering/Grasping -> Lifting (2)
* ab\_angle(1) > ab\_angle(2)
* angle(1) > ab\_angle(2) (Contain above)
* Lifting -> Standing (1) v.s. Lowering/Grasping -> Lifting (2)
* ab\_angle(1) < ab\_angle(2)