**Programming language, compiler, tools, and operating system**
The WAM arm control software runs on Ubuntu 20.04 with a low-latency kernel. We use clang for building our C/C++ code. Some example code is written in Python. We support ROS 1 (Noetic). The computer hardware and operating system are available for sale with the arm, or you may choose to provide them yourself. All of Barrett's operating system configuration, control software, and example code is open-source.

**Haptic Interface**
The WAM arm is a cable driven, fully backdriveable arm. The haptic interface is typically the arm itself, although you may certainly integrate some type of joystick and driver software on the PC to control the arm. Harvard University, Australia National University, and Hong Kong University have driven the WAM arm haptically with a PhanTom haptic interface. The University of Massachusetts owns two 7DOF WAM arms and they use one arm to drive the other (master-master).

**Force control/feedback**
This is a typical control loop:
1) PC asks the WAM for its motor (and optionally, joint) positions.
2) PC receives the position feedback and calculates the forward kinematics.
3) PC generates torques using either the Jacobian Transpose or Recursive Newton-Euler.
4) PC sends motor torques to the WAM.
The WAM arm controls the current (and therefore the torque) at each joint better than any other robotic arm. The system is able to drive any force/torque vector at the end tip of a 7 DOF WAM or any force vector at the end tip of a 4 DOF WAM. The WAM's joint friction is low enough that an external force as low as 100g is enough to move the WAM. This enables force control without the use of a force/torque sensor. If higher sensitivity is required, we offer an optional 6-axis force/torque sensor.
*See force sensitivity document.*

**Positioning tracking**
If integrated with a vision system, the WAM arm can be used to track position in realtime. The high bandwidth of the arm allows high performance in tracking fast movements. Using direct joint torque control, we are able to perform 6-axis position/orientation control WITHOUT calculating the inverse kinematics.

**Controller technology**
Each of the WAM's Brushless DC (BLDC) motors (technically Permanent Magnet Synchronous Motors / PMSM) is controlled by our patented "Puck" motor controller technology. Mounting a Puck directly on each motor maximizes the current control performance while minimizing the wiring bulk and complexity.

**Control rate**
ROS: 250 Hz, typ.
Libbarrett: 500 Hz, typ.
However, these rates are adjustable. Technically, GetPositions = 75uS + DOF * 75uS. SetTorque = 125uS for 4DOF, 250uS for 7DOF. Torque calculation time is usually

negligible. So the highest possible control rate for a 4DOF is 2kHz. For a 7DOF, 1.1 kHz. Adding a force/torque sensor or a BarrettHand will affect the max control rate, as all devices share the same 1Mbps CAN bus.

**Mathematics**

Unlike conventional robots, the WAM arm was designed specifically to exploit the superior benefits of Jacobian-Transpose mathematics in robot control. Jacobian-Transpose control requires *inherent* Joint-Torque control. Inherent means that, even without joint-torque sensors, the joint torques are controlled responsively and precisely. Some robots try to emulate joint-torque control with joint-torque sensors. But, unless the arm is designed for *inherent* Joint-Torque control (only the WAM arm does this), then adding sensors degrades overall performance and, much worse, can cause occasional control instabilities, making them unsafe around people. Very few robots today can claim that they control joint torques, and the few that do make this claim rely purely on sensor feedback and therefore exhibit poor performance.

**Design Principles**

1. Elimination of "Hard" nonlinearities. This means simultaneous elimination of backlash and dry-friction in all parts of all mechanisms. Conventional robots have one or the other, but cannot eliminate both simultaneously.
2. "Transparent Dynamics". Transparent dynamics means that the connection between the motors and the joints is extremely stiff AND that the masses and inertias of all moving components are extremely small compared to similarly-sized robotic arms. It is easy to make a robot arm either stiff or low-inertia, but extremely difficult to do both simultaneously.

**Competitive Advantage**

We believe that it is impossible to achieve this level of performance without using tension-element drives, where Barrett is not only the leader in the field but also holds the key patents. The patent protection explains why other (bigger) robot-arm manufacturers do not try to copy Barrett. Fanuc Robotics, for example, had spent significant effort in the mid-90s to develop a tension-element-driven arm that could exploit Jacobian-Transpose control until they learned that Barrett and MIT own the key mechanism patents.

**Benefits of Jacobian-Transpose Control**

1. The high-stiffness + low-inertia = high speed & high responsiveness (acceleration).
2. Gravity compensation and near-perfect force/torque control.
3. High-performance Haptic interaction (impossible otherwise).
4. Ability to follow any trajectory, even when the goal position is changing in real-time (this is key for vision-control applications). Conventional robots pre-plan their trajectories- attempts to respond in real-time to a changing goal position results in noticeable time delays in conventional robots.
5. Direct Cartesian control of forces/torques and trajectories at the end tip with little computational cost.