Motion Matching for Unity FAQ

Q) Does ‘Motion Matching for Unity’ work with Quadrupeds?
A) MxM can support up to 8 matched joints. Usually only the feet and one central joint (hips) are required. Therefore, it is possible to use motion matching with quads provided you have the adequate animations.

Q) Does this asset require Root Motion to work?
A) You do not have to use root motion to move your character at runtime. However, the animation clips must have root motion in them so that MxM can pre-process them and determine a past and future trajectory.

Q) Will any animation set work with ‘Motion Matching for Unity’
A) This depends on a number of factors. As per the previous section, the animation set must have root motion for the pre-processing stage (not required for runtime) but there are additional requirements to achieve good quality. Animation sets must have good coverage and a lot of transition actions (e.g. walk forward start / walk forward stop etc). The more transitions the better. Motion matching doesn’t like to jump big gaps between animations so this is very important.

Q) Does this asset support other IK system
A) FinalIK and Unity Mecanim IK work right out of the box. Unity’s new Animation rigging package works with an integration script provided. Most IK systems that operate in the LateUpdate will likely work right out of the box.

Q) Does this asset do ‘xyz’ (e.g. vaulting) gameplay mechanics.?
A) Motion Matching is an animation system. It does not implement any gameplay mechanics. However, it can generate animation for any gameplay mechanics so long as you feed it gameplay information and appropriate animations.

Q) Does MxM use machine learning to create animations?
A) MxM does not create animations out of nothing. It uses your animations in a very different way to a state machine though, constantly jumping through your animation library to any pose to try and get the best results. It does this through a more old school kind of machine learning but not Neural Networks. Neural Networks tend to generate approximated animation which can look odd and they are typically not performant enough for real time application.

Q) It seems like this is only good for locomotion?
A) Incorrect, while locomotion is motion matching’s strongest output, MxM ships with an event system and tagging system for things like actions. There’s a layer system and you can even switch between MxM and mechanim during runtime if you feel the need. Motion Matching can achieve almost any kind of animation provided you give it the right inputs.

Q) The animation seems sluggish and unresponsive. What’s going on?

A) MxM can only use the animations you provide it. If the provided animations are sluggish, the results will also be sluggish. I have included a tool to modify the speed of sections of animations to help overcome this but the quality of the source is still a major factor.

Q) My character is hard to control
A) Don't use root motion to control your character movement. This is almost never a good idea in game development in general. There is a hybrid approach where root motion can be used as a trick to improve foot sliding. That will be detailed in an example in the future.

Q) Does MxM handle IK
A) No, motion matching does not handle IK. In realtime animation there are two phases, generating the base animation and the post procedural animation. MxM lives in that first phase just like the state machine. However, IK solutions like Unity's built in IK and third party IK software like Final IK work over the top of MxM out of the box.

Q) Can I use MxM without being able to code?
A) No, you will need to be able to call functions from the MxMAnimator component to control it as an absolute minimum. However, if you want to get the best out of MxM you need to be relatively competent at coding as MxM does not handle gameplay. Playmaker and other visual scripting tools are not supported at release but possibly will be in the future.

Q) Isn't the whole point of motion matching to make things easier?
A) No, the primary point of motion matching is to improve the quality of animations and create player controlled animation that aren't really possible with a state machine. The initial quick workflow is a side effect. In general you get 80% of the way there in 5% of the time compared to state machines but the last 20% requires some work.

Q) Really? No state machines?
A) Really. You may still need a state machine for your gameplay though. MxM does not handle gameplay.

Q) My character doesn't move OR My character just moves around randomly on its own
Before debugging this, turn gizmos on, make sure the MxMAnimator component is unfolded and ensure that the debug pose and trajectory options are checked in the component inspector.

Please confirm the following:

- Do your animations have root motion (you don't need to use root motion to move your character but the animations must have root motion in the source animation for pre-processing)
- Confirm your root motion settings and components (refer to the user manual)
- Check your trajectory generator. Is the Position Bias setting high enough (default 15). Is the max speed high enough?
- Have you set up your character controller properly (if using one)?

- Are you using an MxMInputProfile on your trajectory generator? If so, make sure it is setup correctly.
- Are your animations imported correctly (watch this video)

Q) How can I do ‘XYZ’?
A) Please read the documentation and watch the tutorial videos. If you have done this already and cannot find the answer, ask in the Discord and I will get back to you when I can.