UNREAL ENGINE

Choosing a real-time performance capture system

Contents

Introduction

In recent years, film and television crews have turned to virtual production as a means of enhancing creativity and making impossible shots possible. *Virtual production* refers to a range of computer-aided filmmaking techniques that are revolutionizing the way films are made. From previs and live TV to LED screens that replace green-screen compositing, forward-thinking directors and crews are making use of real-time technology to approach production in a way that was unheard of just a short time ago.

One such technique utilizes *motion capture* (also called *mocap*), the practice of digitizing the motions of a human being or animal for use in analysis or animation. For example, a production team might capture an actor's performance and retarget it to a digital character in real time for viewing in camera.

To explore this booming field, we published The Virtual Production Field Guide to cover all these techniques and more. In this paper we focus on *performance capture*, an extension of motion capture that aims to not only capture the large movements of an actor but also the subtle motions, including the face and hands. By capturing these more subtle details, performance capture aims to recreate the entirety of an actor's performance on a digital character.

Image courtesy of Epic Games, 3Lateral, and Tencent

Figure 1: Performance capture session for Siren Real-Time Performance video [Image courtesy of Epic Games, 3Lateral, and Tencent]

History

In its earliest days, mocap could capture only broad motion, which made it suitable for body motion only. More recently, motion capture systems have become sensitive enough to capture subtle details in the face and fingers, giving rise to performance capture.

Performance capture gives actors the opportunity to use their full range of skills to bring a character to life, and gives a production team a much fuller set of data to realize their digital creations. The recent development of real-time performance capture systems has opened the door to new types of media where an avatar can be driven by a live actor in real time.

Why this paper?

Motion capture and performance capture systems are usually costly, and the time required to acquire and process motion data is not trivial. To get the most for your time and money, it is vital to choose a performance capture system that addresses your production's needs and that will give you the capture data you need without an enormous amount of cleanup or post-production.

This paper addresses the considerations for choosing such a system using examples from recent performance capture projects. The following projects are discussed in this paper:

- Siren Real-Time Performance video released at GDC 2018
- SIGGRAPH 2018 Real-Time Live! Winner: Democratizing MoCap: Real-Time Full-Performance Motion Capture with an iPhone X, Xsens, IKINEMA, and Unreal Engine

While the performance capture systems used on both these projects are real-time systems, the information in this paper is equally applicable to offline workflows.

Types of capture systems

Before you can choose a performance capture system, you will need to understand the various kinds available and how they differ in setup, accuracy, suitability for your needs, and cost.

There are several different types of performance capture systems, which can be categorized by the type of technology they use to capture motion. While there are other types of systems besides those described here, these are the types that are most commonly used for performance capture.

Optical

An *optical* system works with cameras that "see" markers placed on the actor's body and calculate each marker's 3D position many times per second. While simple in principle, these systems require multiple cameras to make sure every marker can be seen by at least two cameras at all times, which is the minimum required to calculate the 3D position. However, the more cameras that can see a marker, the more accurately its position can be calculated.

The markers that a camera sees can be divided into two categories: *active* and *passive*. Active markers are markers that generate their own light that the cameras can detect. Some active markers can pulse light to send additional information to the camera, such as the ID of that particular marker.

Passive markers, on the other hand, do not generate their own light and must reflect it from another source, usually a light ring on the camera. A passive marker doesn't broadcast its identity, so the system identifies each one through analysis of its movement relative to other markers fitted to a mathematical model of the subject being tracked.

One advantage of active markers is that they can sometimes be used effectively with natural light. Conversely, passive markers are impractical for such situations due to the lack of sufficient contrast between the markers' brightness and the ambient light.

As you can imagine, active markers are more robust and can be seen from further away; however, the electronics involved in active markers make them bulkier than passive markers, and their power requirements often require the subject to wear a battery pack.

Due to the large number of tracking cameras required, optical systems can have a high cost and are not very portable. Also, it is worth noting that during live-action photography or a live performance where the actor is visible while motion capture is going on, the visible markers might get in the way of costuming. However, these systems produce the highest-quality results, with accurate positional data and support for multiple actors interacting in the same space along with tracked props and cameras.

Optical systems can capture both body and facial motion. Facial optical systems typically require a head-mounted camera (HMC) system, which is attached to a custom-fitted helmet worn by the actor.

Examples of systems that use optical technology:

- Vicon
- PhaseSpace
- OptiTrack

Figure 2: Mocap suits with optical passive markers used in Epic's demonstration of virtual production techniques

Inertial

An *inertial* system uses miniature sensors called *inertial measurement units* (IMUs) that contain a combination of gyroscopes, magnetometers, and accelerometers to measure the forces and rotations at specific points on the body. The data is usually transmitted wirelessly to the computer, but some systems will record to a dedicated device worn by the actor.

The fact that the sensors do not need to be visible to the camera means that data is transmitted regardless of body position or location behind props or other actors. The sensors can be hidden inside clothing, which makes these solutions a very good choice for situations where the actor will be seen on camera or on stage.

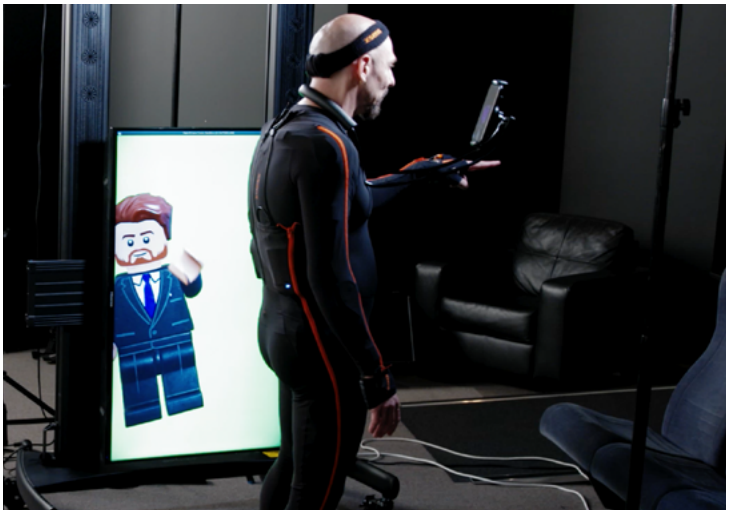

Figure 3: Inertial suit used for LEGO/HQ Trivia broadcast on mobile [Image courtesy of Animal Logic]

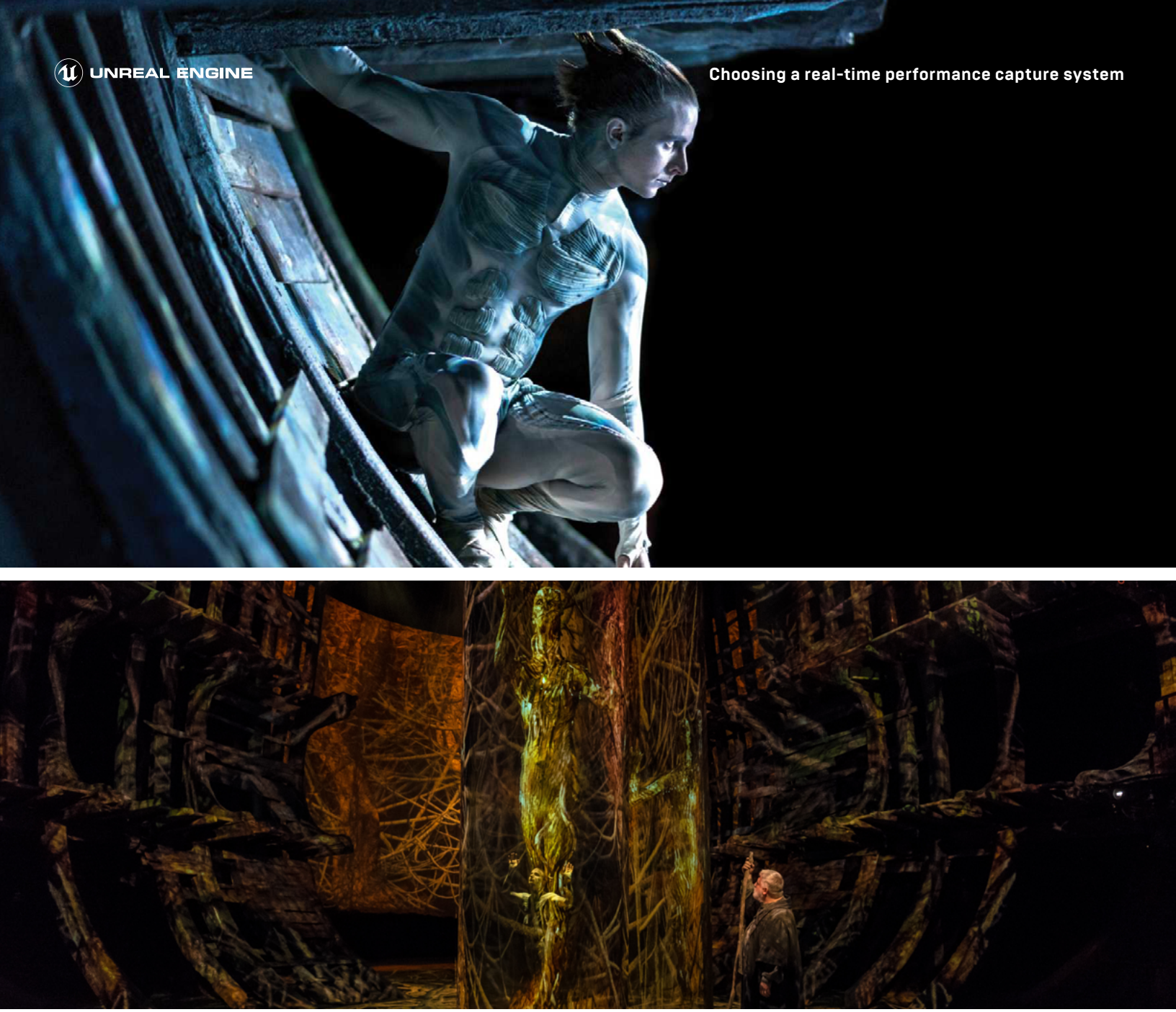

Figure 4: The Royal Shakespeare Company's 2016 production of "The Tempest" utilized real-time motion capture in a stage performance, with Ariel (actor Mark Quartley) wearing a skintight suit with hidden inertial sensors. To represent Ariel being trapped inside a tree, the system projected a 17-foot-tall avatar mimicking his movements in real time. The actor wears the same inertial suit shown in Figure 3, custom-painted for use as a stage costume. [Photos by Topher McGrillis © RSC]

IMUs do not have the ability to know their actual XYZ position in the world; instead, they estimate position based on the motion data they have available. This means that while the pose of the actor is accurately captured, the actor's location in XYZ space is not necessarily perfectly accurate. These types of systems are prone to positional "drift", where the recorded position of the actor can gradually move further away from the actor's actual position over time.

Examples of systems that use IMUs:

- Xsens
- Perception Neuron
- Rokoko
- NANSENSE

Hybrid

In recent years, we have seen the emergence of hybrid optical/inertial systems. These systems retain some of the benefits of each system while removing the drawbacks. One example is the problem of occlusion. If too many optical markers get temporarily blocked from the cameras' views during movement, then there is not enough data to reconstruct the actor's performance correctly. Inertial systems, on the other hand, continue to provide data regardless of occlusion. When combined with the data from the remaining visible markers, this additional data helps ensure that the total sum of the tracking data is solid.

Inertial data can also help reduce jitter (noise in motion detection) when added to data obtained from an optical system. Optical systems have some degree of measurement uncertainty when trying to find the most likely 3D position for that marker on each frame. Even if the marker is not actually moving, this "most likely" position can change slightly on each frame due to various environmental factors such as changes to ambient light. These slight changes make the marker's calculated position appear to shake or jitter. During gross movements this jitter isn't much of an issue as it gets lost amongst the larger motion, but when things are moving slower or a marker is stationary, jitter becomes readily apparent. If inertial sensors are added to the system, when an inertial sensor's accelerometer detects that the marker is moving slowly or stopped, extra filtering can be applied to the optical data to smooth it out and remove the jitter.

It is common for a hybrid system to make use of *pucks*, which are standalone, portable tracking devices. A puck, so named because it's around the size of an ice hockey puck, can be either placed in a fixed location or attached to a mocap suit, prop, or camera. A puck might track inertial or optical data, or both, depending on its internal design.

Figure 5: OptiTrack Active Puck [Image courtesy of OptiTrack]

Figure 6: VIVE Tracker attached to VIVE racket [Image courtesy of VIVE]

There are two main types of hybrid systems that you are likely to encounter:

In-sensor hybridization

In this type of hybrid, the hybridization happens inside the actual sensor. The sensor uses optical tracking to get position and orientation, and also uses IMUs to increase the accuracy of the tracking and prevent loss of data when the sensor's tracking markers are blocked from the view of the cameras.

The IMU provides acceleration and velocity data, which can be used to predict the motion path when the optical tracking is occluded. By using prediction to smooth over occlusions, these type of sensors can be used with far fewer cameras than a typical optical system, which reduces the overall cost. However, the extra electronics required to measure the inertial data make these hybrid sensors quite large.

Examples of systems that use this method:

- VIVE Tracker
- OptiTrack Active Puck

System-level hybridization

This type of hybrid system implements hybridization at the system level. An example of this would be having a full inertial suit providing data and then using one or more optical markers to add extra positional data which can be used to counteract the drift of the inertial system.

While there have been many examples of this approach being used, it can be very tough to get a good fusion of the two data sources. In theory, you could, for example, add a marker to the actor's hips and then set the position of the inertial suit's hip data to be the marker position. However, no matter where you place the marker on the actor's body, it will not move rigidly with the actor's pelvis due to the way human hips work. When the actor bends forward, for example, the marker is likely to move up or down in space in a way that doesn't reflect what the pelvis is actually doing. This can lead to data that makes the target skeleton appear to crouch (or even levitate) when the actor is simply leaning or bending over.

Example of system that uses this method:

- Xsens inertial suit + VIVE Tracker for location

Image-based

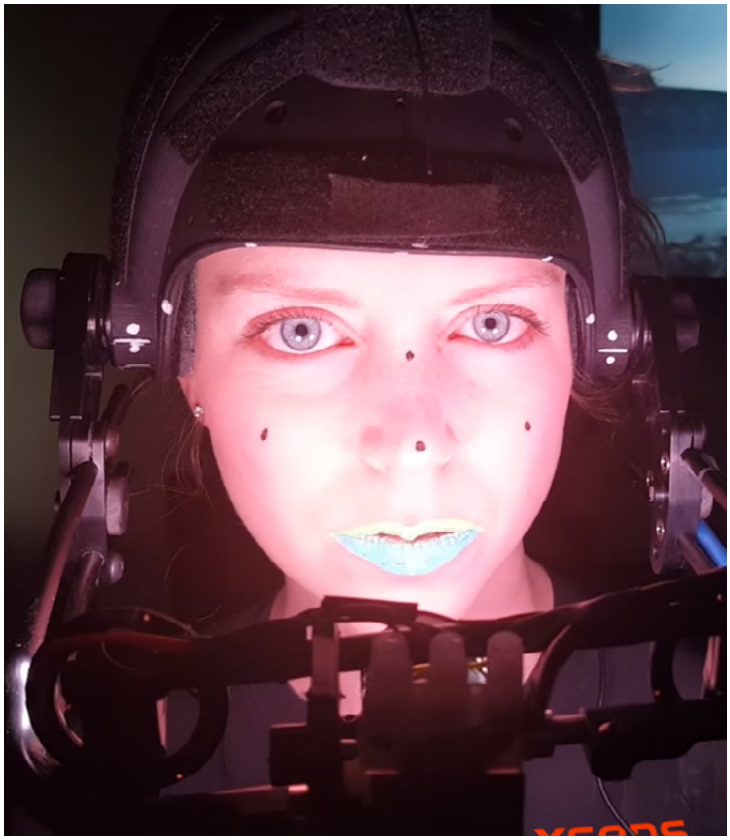

Figure 7: Head-mounted camera
[Image courtesy of Ninja Theory, Cubic Motion, and 3Lateral]

With an image-based system, the two-dimensional images taken with one or two cameras are used to interpret the changing three-dimensional shape of an object over time. As a familiar example, the Kinect camera uses this system to provide a somewhat crude but inexpensive motion capture system for non-professional purposes.

For professional purposes, image-based solutions are generally suitable only for facial capture. When cameras for an image-based system are attached to a helmet worn by the actor, the cameras move with the character's head to always point directly at the face. With such a relatively small area needing to be captured, and the limited range of possible expressions in the human face, this type of system can provide a sufficient level of detail in facial movement for transfer to a digital character in real time.

Using an image-based system for tracking the body can still be useful in cases where you do not need a fully accurate model of the person's body but just a sense of what that body is doing at that moment.

For example, you might not need to know precisely where an actor's arm is, only whether it is moving quickly toward an object. This type of capture is particularly useful in installations where you want to respond to user motion without forcing the user to wear anything special for the tracking. An example would be an educational installation at a museum where you want passersby to be able to use gestures to interact with the content without having to do any kind of setup or calibration beforehand.

Professional image-based systems for facial capture usually come with additional tools for processing the actors' facial movements and converting the raw data into higher-level information. In addition to this raw data, you might also get, for example, a value that represents how much the mouth is smiling, or perhaps how open the mouth is. These higher-level abstractions are normally represented as *blend shapes* (also sometimes called *morph targets*). Blend shapes are often used with an additional transform to represent the head rotation.

Figure 8: Facial expression and associated blend shapes' percentages

Figure 9: Face mesh driven in real time by iPhone X camera

To make an image-based system work with your target character, you will likely have to go through some sort of mapping process to take the incoming blend shapes and format them in a way that will work for that character. Figuring out the exact blend shape mapping for a particular avatar mesh can be tedious, as it requires manual work. However, once this mapping is found, you can reuse it with any source actor—the system will output the same set of shapes for the target character no matter which actor is being captured.

Once you have experience with a particular image-based system, you can build future characters with facial rigs designed to work with the system, which will minimize this mapping time.

Examples of image-based systems:

- Cubic Motion Persona HMC
- Apple ARKit face tracking

Determining your needs

Pose versus position

One of the most important decisions you will need to make has to do with the type of data you need to capture, which will in turn affect your setup. There are two major setup / data types: *pose* and *position*.

A *pose* system captures the actor's motions relative only to himself. A *positional* system captures the actors' and cameras' motions relative to a fixed location in space, and from this data is able to determine each actor's position relative to other actors and the camera.

A positional system captures each actor and camera relative to an XYZ origin coordinate location (0,0,0) somewhere in the space, while the pose system captures each actor in relation to a fixed location somewhere in the actor's pelvic or hip area.

For each capture session, you will need to decide which of these you are going to use. The reason for making this distinction is that a pose system generates less complex data and is easier to set up than a position system. If you can get away with using a pose system, you should do so.

You can use a pose system if your project meets *all* the following criteria:

- During capture, actor(s) will not physically interact with other actors, or with tracked props.
- You are not concerned with tracking the camera for the background behind the character(s), for example:
    - You are planning to have the camera static during all shots and are happy to match up the background manually in post, or
    - The background is a plain color that will not change, even if the camera moves.

An example of a situation where pose capture is acceptable is where you plan to capture two actors dancing with each

other but not touching, and you will be switching between static camera views in the final rendering. Additionally, if one actor/character is holding something such as a cup, it will either need to be "locked" to the hand, or would have to be added later in post.

You need a positional system if any of the following are true:

- You have multiple characters being captured together and they need to interact (touch) by shaking hands, fighting, hugging, etc.
- At least one character is interacting with a tracked prop during capture.
- The camera will move during shots, and you need to track the camera and/or background for lining up later with the captured action.

A positional capture session requires the use of an optical or hybrid mocap system to ensure everything lines up. Conversely, a pose capture session can use an optical, inertial, or hybrid setup.

If you can slightly alter your script or action to switch from a positional to a pose system without affecting the quality of your story, it's worth taking a look at doing so—it will give you more options and potentially save time and money.

As an example, we used a positional system for the Siren project as we wanted to know where the character was within the space as she moved. The Kite and Lightning project, on the other hand, was able to use a pose system because the characters didn't touch each other or move around a great deal. We will discuss these choices in more detail later in this paper.

Figure 10: Motion retargeted from a positional capture as the actor walks [Image courtesy of Epic Games, 3Lateral, and Tencent]

Facial capture

If you plan to capture facial expressions or speech, optical and image-based systems are the only two real options. Inertial IMUs are not suitable due to their large size versus the small scale of motion they would need to measure.

An optical system can provide cleaner results than an image-based system, but it usually requires the application of some type of facial markers to the face. Physical markers can take a long time to set up, and are prone to falling off during lively facial movements, so the more common method is to draw markers onto the face to provide additional features for the system to track.

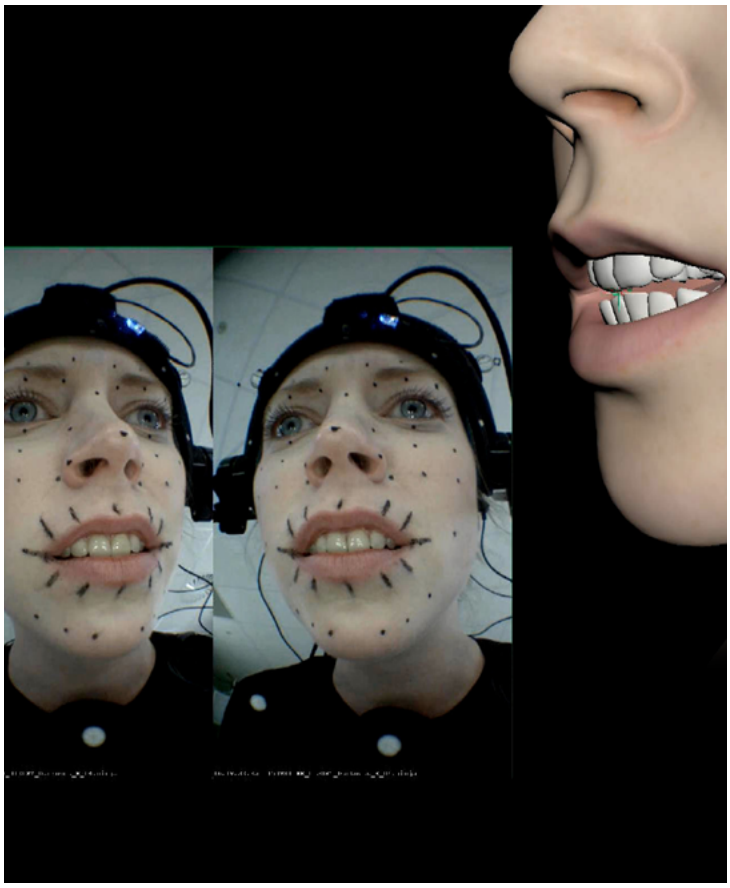

Figure 11: This figure shows markers drawn on actor Melina Juergens' face, and the resulting animation targeted to the Senua character, for the Ninja Theory game "Hellblade: Senua's Sacrifice". 3Lateral used its proprietary 3D scanning system to capture Juergens' appearance and build a digital double facial rig, and 3Lateral's Rig Logic technology enabled the rig to run in real time. Then Cubic Motion's real-time facial animation technologies brought Senua to life. [Image courtesy of Ninja Theory, Cubic Motion, and 3 Lateral]

Figure 12: Doug Roble of Digital Domain using head-mounted camera for facial capture [Image courtesy of Digital Domain]

More commonly, real-time facial capture systems make use of a helmet with one or more rigidly attached cameras (head-mounted cameras, or HMCs) hanging in front of the actor's face. It is worth noting that the hanging cameras can be distracting to actors to the point where it interferes with their performances, especially for actors new to the process. However, if you're going to do real-time facial capture where the actor needs to move around, this is the way to go.

For real-time facial capture where the actor is static, or if the data is being captured as part of a non-real-time workflow where the actor's facial motion will be applied to a character later, then you might be able to use a helmetless facial capture system. An example of such a helmetless solution is Disney's Medusa system.

You should also take into consideration whether you plan to capture speech, or just facial expressions. In general, capturing speech requires your system to be much more sensitive than if you are capturing just expressions.

Hand and finger capture

You will also need to decide on the level of accuracy you need for capture of hand and finger motion. With most mocap systems, you can get a reasonable degree of accuracy for basic wrist movement and rotation, which is often enough for simple character activities like walking or talking. However, capture of individual finger motions is typically more challenging.

If a character grabs a prop or the fingers take on a movement where positioning is important (pointing, stroking a beard, etc.) then you will need to consider finger capture unless you are prepared to invest some time in manually animating the fingers. If you've already got your eye on a system for body capture, find out whether it supports finger capture, and if so, the types of values it captures. For example, some systems will only provide curl values for fingers and not the spread.

Figure 13: The Manus Xsens Glove combines IMU sensors from an Xsens suit with specialized finger sensors to provide accurate hand tracking data that correctly integrates with body data. [Image courtesy of Manus VR]

If the system you are considering doesn't have support for finger capture directly, there is a range of separate systems that can be used along with your primary system to capture this data. These separate systems usually take the form of gloves, and generally either use IMUs or strain sensors to measure the positions of the actor's fingers. The finger data is then merged with the wrist data coming from the primary mocap system.

Budget

There are three main routes to go with your performance capture budget: contracting with a service provider, renting a system and running it in-house, or purchasing a system for in-house use.

Contracting with a service provider is a good choice if you haven't done performance capture before, or if the equipment you want to use is complex. A service provider will help you choose a good system for your needs, provide the space and personnel to run the session, and deliver the data you require. Even if you plan to do performance capture regularly, such a session (or a series of sessions) can be very helpful in giving you greater knowledge and understanding of what you'll need when you eventually acquire your own system.

Renting a performance capture system for in-house use makes sense if you think you're only going to need motion capture occasionally, and you know what you'll need. You'll also need to have 1-3 technicians to set up, transfer, manage, and clean the mocap data. One advantage of renting over owning is that you won't have to maintain the equipment.

Purchasing a performance capture system could be the way to go if you plan to use it enough to justify the expense, know exactly what you want, and have sufficient personnel to run and maintain the equipment.

There are no set rules or prices when it comes to purchase or rental of performance capture systems. Prices go up and down regularly based on demand and other factors.

Here is a brief comparative guide to the cost of purchasing a performance capture system, as of the time of this writing:

- Tracking puck-based system - $
- Budget inertial suit - $
- Hybrid system - $/$$ (Depending on inertial suit chosen)
- High-end inertial suit - $$
- Rent time at professional mocap studio - $$ per day
- Build optical studio - $$$$

How many actors at once?

The number of actors you need to capture at once will likely be a big determining factor in the type of system you choose.

In general, an inertial system is the least expensive "starter system" for body motion capture of a single actor. However, if you need to capture several characters at once, purchasing several inertial suits can end up being more expensive than an optical system.

	Inertial	**Optical**
1 actor	$	$$$$
2-3 actors	$$	$$$$
4-5 actors	$$$$	$$$$
6+ actors	$$$$$$$	$$$$

While the cost of an optical system is initially higher, the cost to add an actor is much lower.

For those needing to capture several actors on an extremely low budget, you do have the option of purchasing a single inertial suit and then capturing each actor individually. This option is viable only if you are able to use a pose system, where the actors don't touch each other and the background or camera does not need to be tracked. Such an approach does add more work in post-production, but it saves money on the initial outlay for motion capture gear.

Case studies

To help guide you in your choice of performance capture system, here are some examples of recent projects. Each had different needs, and each utilized a different solution to address the project at hand.

Siren

In the Siren Real-Time Performance video shown at GDC 2018, Epic Games created a digital human named Siren to be driven in real time by a motion capture actor. The actor's facial and speech motions, voice, and body motions were captured live and targeted to Siren in real time.

Figure 14: Siren character driven live by actor, including speech and basic hand motions [Image courtesy of Epic Games, 3Lateral, and Tencent]

To prepare the demo, we started out by listing our needs for performance capture:

- Position would be important. We wanted to use a moving virtual camera, which meant we would need to track both the camera and actor to ensure everything lined up.
- We wanted facial capture, as Siren had to be able to talk to the audience during the piece. However, we didn't want to use markers on the actor's face since this was going to be a live demonstration, and we couldn't risk markers falling off during the presentation. This left us with the option of using an image-based system for the face.
- For the body motion, however, we were okay with having visible markers on the body, as we knew we could affix them strongly enough to the mocap suit to withstand the rigors of live presentation.
- We also knew that we wanted Siren's fingers to move. However, we did not need full detail, just the ability to have natural movement while the live actor was talking.

Because position was important and we weren't constrained by having to hide the markers, we went with an optical marker-based system for the body, in this case Vicon. For facial capture, we had multiple options for camera tracking, however we went with Cubic Motion's real-time facial capture solution due to already having familiarity with their system.

For finger motions, since our needs were modest, we were able to use Vicon and thus avoided adding another motion capture system. We used Vicon's Shogun software to track key points on the fingers and solve them into plausible finger positions.

SIGGRAPH 2018 Real-Time Live! Winner by Kite & Lightning

Los Angeles-based Kite & Lightning worked on several VR projects before experimenting with real-time motion capture. Cory Strassburger, one of the company's founders, wanted to take some of the baby characters from their game *Bebylon Battle Royale* and develop a system where he could drive the characters with real-time mocap. The resulting demonstration, Democratizing MoCap: Real-Time Full-Performance Motion Capture with an iPhone X, Xsens, IKINEMA, and Unreal Engine, took home the winning prize at the SIGGRAPH 2018 Real-Time Live! Competition.

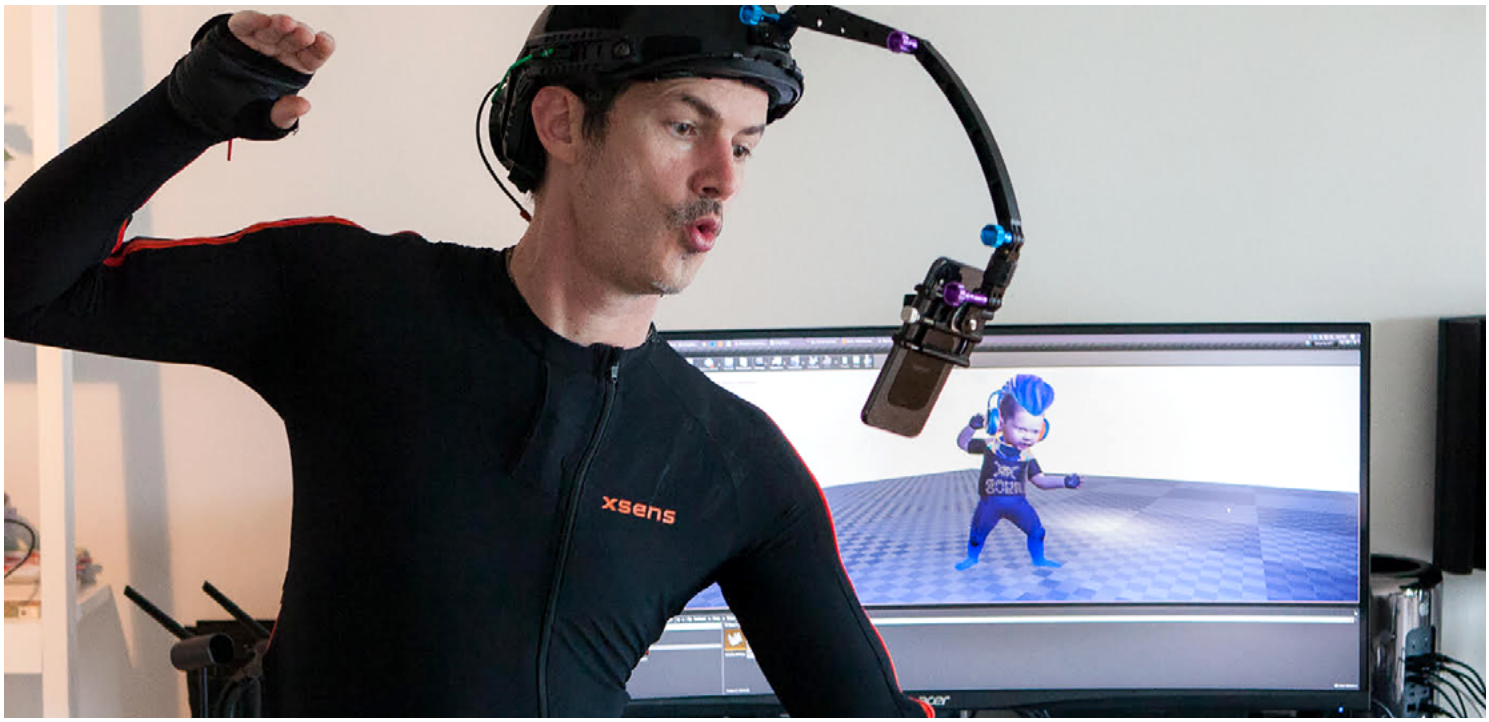

Figure 15: An iPhone X acts as a depth sensor/camera for facial capture. [Image courtesy of Kite & Lightning]

For body capture, a pose system was sufficient since the characters appeared against a static background, and didn't touch each other or walk around a great deal. Cory wore an Xsens MVN inertial suit, and used IKINEMA LiveAction (now part of Apple) to transfer the streamed body data from the suit to Unreal Engine.

For facial capture, he attached an iPhone X to a paintball helmet and utilized Apple's ARKit to stream the facial capture data into Unreal Engine. You can learn more about this implementation on this Unreal Engine blog post. Digital studio Animal Logic used a similar approach to capture a performance and retarget it live to a LEGO minifigure, which you can also read about on another Unreal Engine blog post.

For full documentation on how to use the iPhone X's facial system in Unreal Engine, refer to the Unreal Engine Face AR Sample documentation.

Summary

Choosing a performance capture system can take time and effort, but is well worth it when you find a system that fits your needs and budget.

Keeping in mind the features and limitations of position, pose, facial capture, and finger capture systems, and the benefits, drawbacks, and price considerations of renting versus owning a system, you are now in a position to make intelligent choices for your own performance capture projects.

The future of performance capture

With real-time performance capture becoming more accessible and accurate than ever before, we're excited to see its increasing use in films, games, and other media where the subtleties of human performance are needed in real time.

Forward-thinking companies like Digital Domain continue to develop, research, and test new performance capture systems like the one for Digital Doug, a real-time animated counterpart to Doug Roble from DD's R&D department. And here at Epic, we are constantly striving to develop new technology to ensure that as new and innovative performance capture systems and use cases (such as Digital Doug) emerge, Unreal Engine will be there to support your creative vision. One example is Live Link, Epic's built-in plugin for ingesting performance capture data into Unreal Engine, which works with a wide range of systems and software. We are also continuing to improve our support of timecode and genlock within Unreal Engine to ensure the most accurate data possible for both real-time and recorded performance capture data.

While every effort was made to make the information in this paper accurate, it's important to note that the world of performance capture is still evolving and continues to change almost daily. We are often surprised by new technology or hybrid implementations that attest to the ingenuity of developers in this field. We are just as surprised (and pleased) when we see that these systems, once the sole domain of large studios, are now more accessible to a much wider range of projects and budgets.

We hope you will follow along as performance capture systems continue to become faster, more accurate, less costly, and easier to use.

Resources for further learning

Further reading, videos, interviews: Unreal Engine Virtual Production Hub

Video: Real-time Motion Capture in Unreal Engine

Video: Siren Behind the Scenes

Blog post: Unreal Engine helps power Kite & Lightning's winning performance at Real-Time Live!

About this paper

Author

David Hibbitts

Editor

Michele Bousquet

Layout

Jung Kwak