Vegard Myklebust Showreel 2013 Breakdown and Technical Descriptions of techniques used.

Programming languages: C / C++, C#, Python, Boo.
Programs: Unity3d, messiah:Studio, XSI, Lightwave, Modo, 3d Coat, Marvelous Designer, Fusion, AfterFX, Photoshop,

Shot 1: Title Card

Shot 2: Model / Main texture maps created by Infinite Realities LTD.
I did the rig, animation and shading / shader technical setup.
Techniques used:

- Vertices are delayed based on a weightmap to simulate inertia in fleshy areas away from the bone structure.
- Animated object space weights tied to the rig define areas of influence where colour changes should occur.
- The same technique was used to define areas where texture and displacement maps would be blended and surface properties changed.
- By having the weights animated in object space we can blend in smoothly over an area without uniformly doing so over the whole weightmap and the number of maps to achieve different areas of blending is reduced. The animated weight can expand in all directions, change strength and falloff. The animation is easily connected to the rig for automation.
- Additionally some displacement maps sit in the space defined by a non deforming stretchy bone (as opposed to UV space) to simulate muscles and tendonds sliding under the skin.

Shot 3: Walker - Automated walk cycle generator

- The system 'Walker' inputs one or more walk cycles that have been animated with a generic rig.
- Walker then parametrisizes the positional information of the hip motion by comparing the height of the hip from the ground and the width of the hips from one rig to another. This helps naturally scale the motion to any sized character.
- Locators are placed by the animator to position the steps of the character.
- The position of the feet are faded into and out of the footstep locators based on time multiplied by an animatable speed value and when the foot is not on the floor it gives respect to the keyframed animation.
- The rest of the body motion is also based on time * speed and controls the blending of the motion additively over the rest of the keyframed motion.

Shot 4: PLANKTON INVASION (c) 2012 – TeamTo – Vivi Film – Tinkertree
For Plankton Invasion I created a system called EPIC (Extreme Plankton Invasion Code) to do the large number of effects in the show. The system was written in python and was developed agile so that we would have a system that worked from the start and added effects gradually as they turned up in the episodes.

- A shot would be tagged in a breakdown database to have a certain selection of effects applied to it.
- EPIC would look up the shot in the database, and then create the necessary 3d passes and scenes based on the animation scene.
- It would then submit the necessary passes to the render farm.
- Further, EPIC used artist created preset comp files for each effect, and created a

compositing file where all the necessary elements were put in place, all the correct image sequences were automatically loaded, and shot lengths set. It had a priority system for determining which effects should normally go on top of eachother in the comp.
- Using a locator object in Maya, the 3d artists working on the shot would be able to time when and where for example an explotion should occur, and thus the system had enough information to prepare a UV pass in that area for use in compositing and time the effect.
- A very common effect was sand from the feet of the characters, and this was calculated based on the animation curves of the characters by EPIC so that it would not be necessary to place locators or particle effects in the 3d scene by hand.
- In the end of year 1 over 60 effects were fully or partially automated and the system enabled 5 artists to render and composite 6 x 7 minute episodes of the effects heavy show per month with up to 50% of shots being effects shots.

Shot 5: 850 meters short film. (c) 2010-2013 Thuristar – Lunanime
My work included setting up the character rigs and some of the same rendering techniques for animated displacement maps described under Shot 2.
I also created the pipeline software for combining the animation shots and proxies with the render ready elements. Collaborating with Joeri Christiaen, the animator and director of the movie, I created a number of smaller scripts and animation tools to enable Joeri to work even faster.

Shot 6: Soft Contact Deformer
- Implicit objects are used to create a fast inside/outside and closest point test.
- Vertices are marked as collided or not in an array, and depth of collision.
- Vertices that are within AABB of the range of the collided vertices are tested for distance to closest colliding point.
- Distance is mapped to the animator defined curve which controls the surrounding area.
- Sphere, Tube and Spline colliders have additional efficiency approximation mode where instead of distance to all collided vertices, distance to the implicit object is used to approximate closest collision, which also gives smoother results.
- Implemented in C for fast execution. Prototyped in XSI ICE

Shot 7: Time independent Jiggle Deformer
- Initial one time calculation yields distances of vertices to a locator in rest position.
- By computing the matrix of the locator at several different times and interpolating between them, the locator can offset its motion in time onto the vertices based on distance, or weight map values.
- Very animator friendly, single control for a lot of nice overlapping motion.
- Implemented in C and prototyped in XSI ICE.

Shot 8: Procedural Animation / Smearframe Deformer
- Proprietary Procedural animation system.
- Uses tricks inspired by stock-in-trade animators to speed up animation process.
- Smearframe deformer delays vertices based on weights and/or their normal direction compared to that of their speed direction.

Shot 9: Some of the very many rigs I've done over time.
- Rig 26 displaying simplified on face controllers for direct manipulation as well as

box controllers.
- Rig 32 displaying IK curve over FK chain for good shape control in animation.
- Rig 95 displaying simple yet flexible eye controllers that are fast to use in animation.

Shot 10: Rigs / Animation for Ad – Property of Nile TV / Darkside Animation
- Novel technique used to sculpt vertices in time while going between two morph shapes to create the effect of several portions of the mesh turning into smaller characters.

Shot 11: Grandmother's Footsteps
Grandmother's footsteps is a game that I created in it's entirety. Models, texture maps, rigs, animation, game design and all code including shaders. The game is made in the popular game engine Unity. It is available on iOS and Android with Leap motion control for PC and Mac coming soon. The music was composed by Hollywood based composer Haim Mazar.

Shot 12: DangleSmash:
DangleSmash is another game that I created in Unity. This time with help from Peet Lee to help create the main 3d models, texture maps and sound effects. I created all the rigs, animation, code and everything else in the game. The music was sourced from public domain russian war marches.

Shot 13: Iron Maiden Music video Property of Iron Maiden / Darkside Animation
On this Iron Maiden music video I created all the rigs, and some of the animation as well as doing the technical setup of some of the shaders.

Shot 14: Title Card