
# SMFoLD

## Streaming 3D Media

C. E. (Tommy) Thomas, PhD

Paul Jones ◆ Steve Kelley

Third Dimension Technologies

SMFoLD Workshop

October 3, 2017

# Standard for Streaming 3D Media

- Sponsored by the Air Force (AFRL)
  - Phase I Completed
  - Phase II Initiated September, 2017
- Program Facilitators
  - Third Dimension Technologies
  - Oak Ridge National Laboratory
  - Insight Media

SMFOLD.ORG

# Third Dimension Technologies

- TDT is Developing Electronic Holographic Stereography Displays
- Current Projects
  - Flight Simulator with Integrated 3D Light Field Display
  - Standard for Streaming 3D Media to Field of Light Displays (SMFoLD)

3RD DIMENSION TECHNOLOGIES

# Oak Ridge National Laboratory

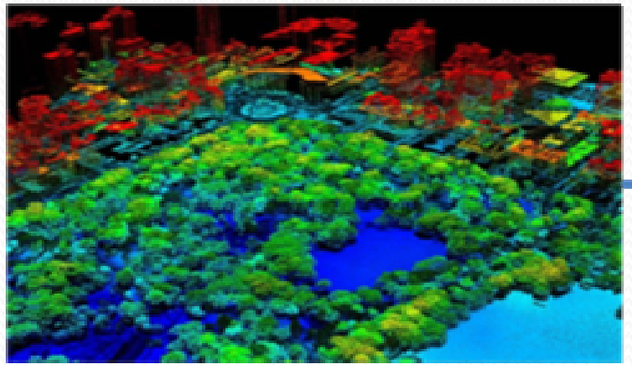

- ORNL Leadership Computing Facility (OLCF)
- Computer Graphics Experts
  - Jamison Daniel
  - Benjamin Hernandez, PhD

OAK RIDGE National Laboratory

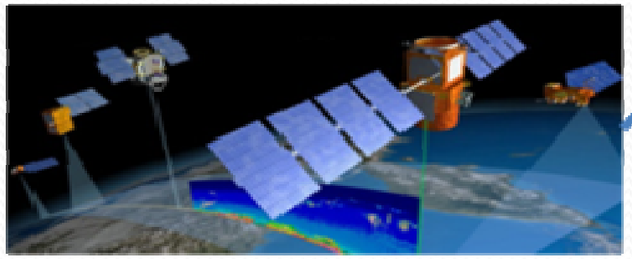
# AFRL Identified Issues


- 3D Sensor Data Increasing Dramatically
  - LiDAR, SAR, plenoptic camera, stereo or multi-view to 3D
  - 3D models (actual and created)
- 3D Visualization Needed to Improve Productivity
  - Stereoscopic 3D (S3D) not acceptable
  - Field of Light Display (FoLD) is desired
- Lack of Streaming Model is Barrier to FoLD Adoption
  - A standard is required so that all software applications send 3D streaming data to SMFoLD displays in a non-proprietary standardized format

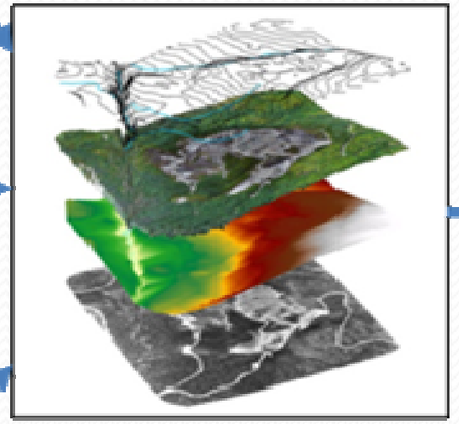
# SMFoLD Streaming Application



Enormous Amounts of 3D Data Being Generated

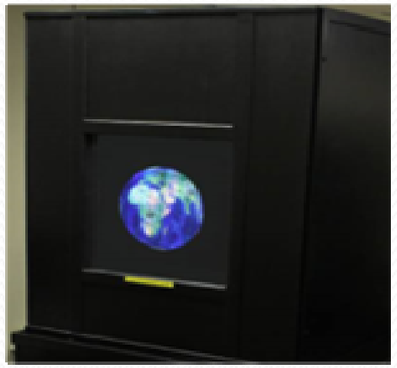


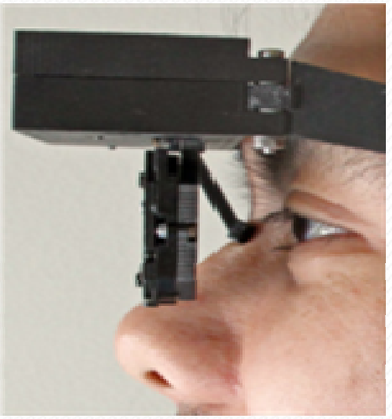
3D Streaming?



Spatially Heterogeneous and Multi-Scale Data with Inherently 3D Context

New Class of Field of Light Displays (FoLDs)





# AFRL Objectives


- Display Agnostic 3D Streaming Media Model
  - Specifically for Field of Light Display (FoLD)
  - Viewable on any 2D, S3D or FoLD system
  - Flow and view parameter control
- Creation and Briefing of Technical Report
- Workshops to Facilitate Process

# Field of Light Display (FoLD)

- No Vergence-Accommodation Issues
- Horizontal and *(sometimes)* Vertical Parallax
- Images Perceptually Indistinguishable from Reality *(ideally)*
- Standalone FoLD Systems or FoLD Eyewear
- Commercial, Military and Government Applications
- FoLD System Types:
  Holographic, Integral-Ray (Hogel), Stereographic, Volumetric

# 3D Data Sources / Types

- Light Field Data
  - Plenoptic Cameras
  - Camera Arrays
  - Moving Cameras
- Depth Maps (x,y,RGBD)
  - Time of Flight (ToF) Cameras
  - Structured Light Cameras
- Point Cloud
  - LiDAR
- Object Representation
  - Synthetic Aperture Radar
- Digital Formats
  - 3D mesh with/without textures
  - CAD data
  - Planar primitives
  - Voxels
  - MRI/CT slice data

# Some 3D Standards and Activities

- ISO/IEC MPEG
  - 3D-HEVC (multiview + depth)
  - MPEG-I-Visual
- SMPTE
  - 3G SDI (Stereo 3D)
  - ST 2087 – Depth Map Representation
- JPEG PLENO
  - Open standard in progress
  - May not support all display types
- Chromium
  - OpenGL 3D data over TCP/IP
- WebGL
  - OpenGL wrapper for browser 3D
- Open3DGC
  - MPEG 3D graphics implementation
  - Khronos Group – glTF
- Virtual Reality Industry Forum
- W3C – Open web platform for VR delivery

# SMFoLD Objectives

- Develop Display Agnostic 3D Streaming Media Model
  - Includes scene description and transmission format
  - Allows for flow and point of view control, other variables TBD
  - Visualization on any 2D, S3D or FoLD system
  - Open standard that supports DoD needs
  - Easily implemented by 3D software application developers
  - Supported by FoLD system developers

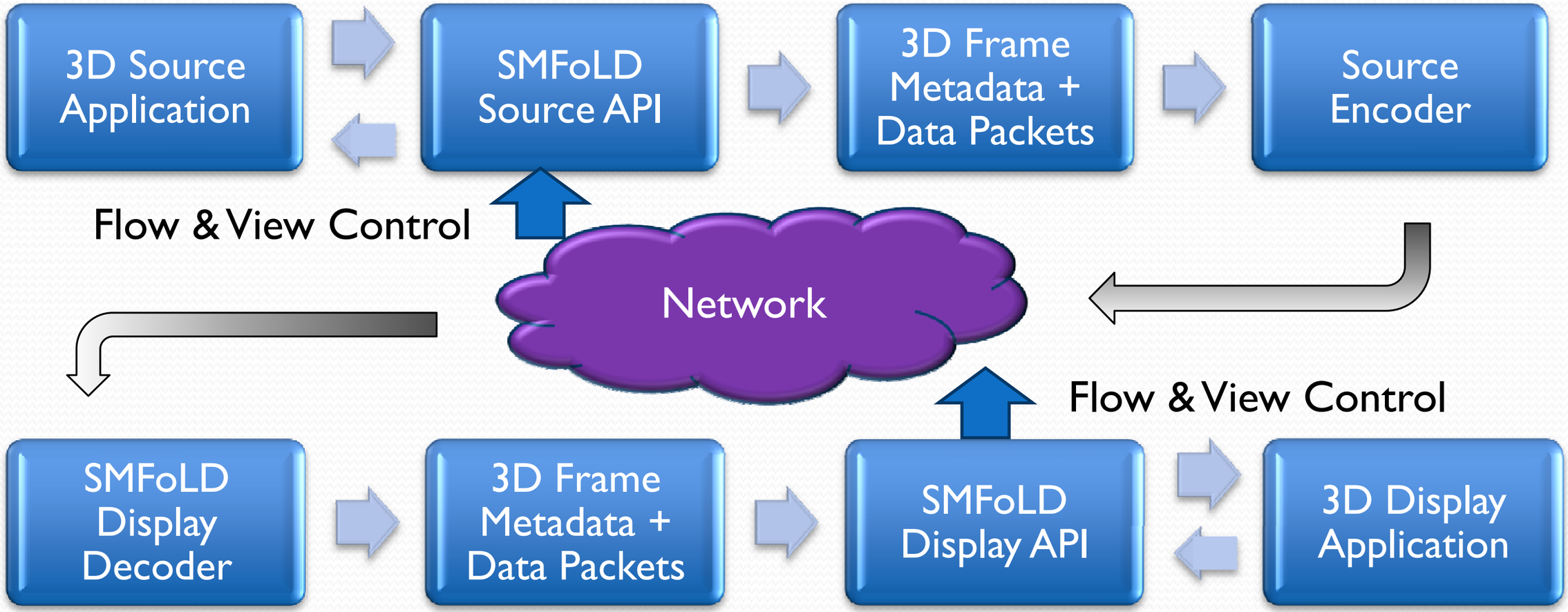
# Problem to be Solved

- Light Field Displays Must Render 3D Scenes from Multiple Viewpoints
  - 3D applications don't provide all of the information needed
  - Information in precompiled shaders is inaccessible
  - Many data types and graphics rendering interfaces

# TDT's Start Simple Solution

- Use OpenGL for the SMFoLD Interface
  - Source and Display Applications Link to SMFoLD Library (SMFoLD.dll)
- Data Types Limited to Mesh, Texture & OpenGL Primitives
- Hooks to Named Variables in Vertex Shaders
  - Camera position, camera angle, focal plane, camera field of view, and others (TBD)

# Proposed SMFoLD Flow Model



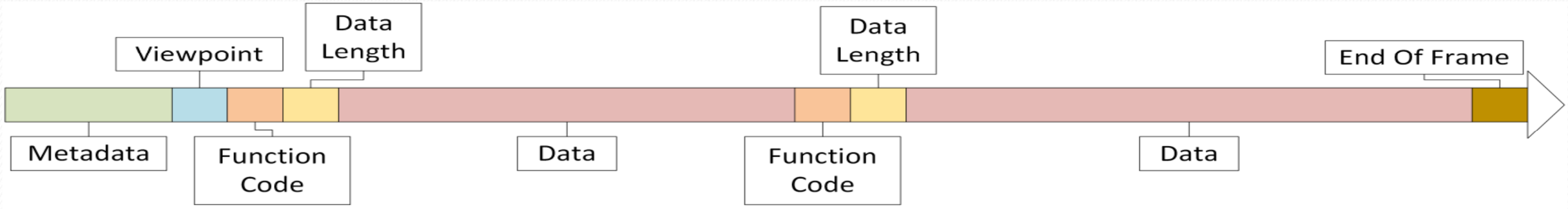
# What is in an SMFoLD 3D Frame?

- A 3D frame contains all of the information needed to display an image on a 2D or 3D display
  - Values that represent function calls
  - Data that the function calls use
  - Metadata to allow the display to create any number of viewpoints
- 3D objects are frequently defined as discreet objects
  - Object definition downloaded once and stored locally
  - Object can be locally manipulated without changing the object data

# What is in an SMFoLD 3D Frame? (2)

- Graphics primitives are functions that are used in an application to describe a 3D scene
  - Viewpoint as defined by the application
  - Metadata needed by 3D displays for multi-viewpoint rendering
  - Geometry transformation matrix
  - Colors, material properties, blending, etc.
  - Arrays of values expressing 3D structures or models.
- Shaders provide the rendering pipeline logic for all frames

# Typical SMFoLD 3D Data Frame Format

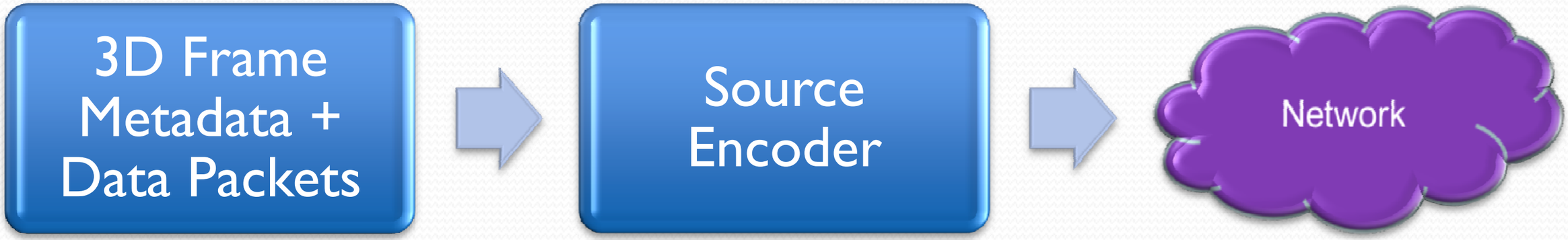



**Structure of the SMFoLD Stream**

# API Encodes 3D Frame


- Application Generates 3D Data
  - Mesh data with or without textures
  - Shaders with hooks to give display access to variables
- Application Calls API to Create a 3D Frame
  - Graphics primitives describe scene
- API Encodes Functions & Arguments, Writes to Memory Buffer
  - API calculates the required Metadata as 3D frame is created

# 3D Frame Encoded




- 3D Frames Compressed and Encrypted
  - MPEG open-source, royalty-free mesh compression (3DMC, BBA)
  - glTF transmission container includes compression technology
- 3D Frames Output to Network
  - Bandwidth required for transmission?
    - 3D frames (mesh, textures, shaders, metadata) not necessarily huge

# Bit Rate/Frame Rate vs Compression

| Example Application | Frame Rate (FPS) At % Compression (C) Network Speed 1Gbps (125MB/s) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Frame Size (Bytes) | C 25% | FPS | C 50% | FPS | C 75% | FPS |
| Google Earth | 2,049,837 | 1,537,377 | 81 | 1,024,919 | 122 | 512,459 | 244 |
| Poles | 1,248,427 | 936,320 | 134 | 624,214 | 200 | 312,107 | 400 |
| 3D Fish | 4,279,805 | 3,209,853 | 39 | 2,139,902 | 58 | 1,069,951 | 117 |
| QT Reader | 6,705,819 | 5,029,364 | 25 | 3,352,909 | 37 | 1,676,454 | 75 |

# Bit Rate/Frame Rate vs Compression

| Example Application | Frame Rate (FPS) At % Compression (C) Network Speed 72Mbps (9MB/s) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Frame Size (Bytes) | C 25% | FPS | C 50% | FPS | C 75% | FPS |
| Google Earth | 2,049,837 | 1,537,377 | 6 | 1,024,919 | 9 | 512,459 | 18 |
| Poles | 1,248,427 | 936,320 | 10 | 624,214 | 14 | 312,107 | 29 |
| 3D Fish | 4,279,805 | 3,209,853 | 3 | 2,139,902 | 4 | 1,069,951 | 8 |
| QT Reader | 6,705,819 | 5,029,364 | 2 | 3,352,909 | 3 | 1,676,454 | 5 |

# Compression Approaches

- Several Techniques—Experimentation Required
  - Lossless
    - Run-length encoding (textures)
    - LZ or LZX (basis for ZIP files)
    - Grammar-based
    - 3D Point Cloud Compression (De Queiroz)
    - MPEG 3DGC (Connectivity) (royalty free)
  - Lossy (converting floats to integers or lossy compression of textures)
    - Allows for some information loss
    - Trade offs between preserving data and size
    - JPEG is an example of lossy compression
    - MPEG 3DGC (Geometry) (royalty free)

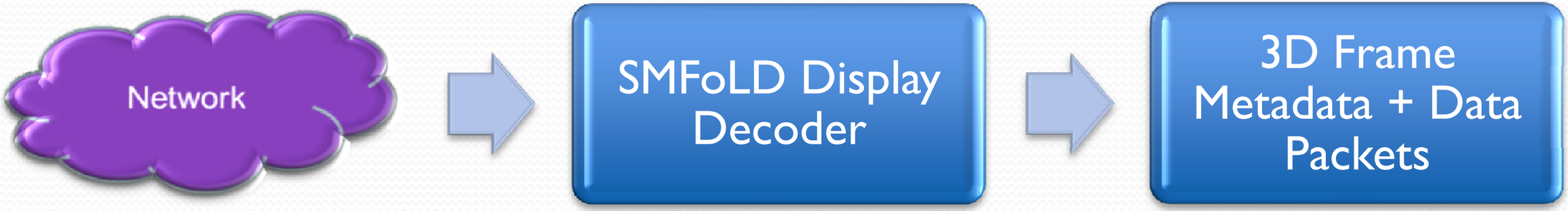
# Mesh Compression Approaches

- May Use Along with Full Frame Compression
- May Introduce too Much Latency for Some Methods
- Various Mesh Compression Methods
  - Edge Breaker
  - Cut-Border Machine
  - Quantization and Prediction (convert to integers--may be relatively low latency)
  - Progressive Compression

# Encryption

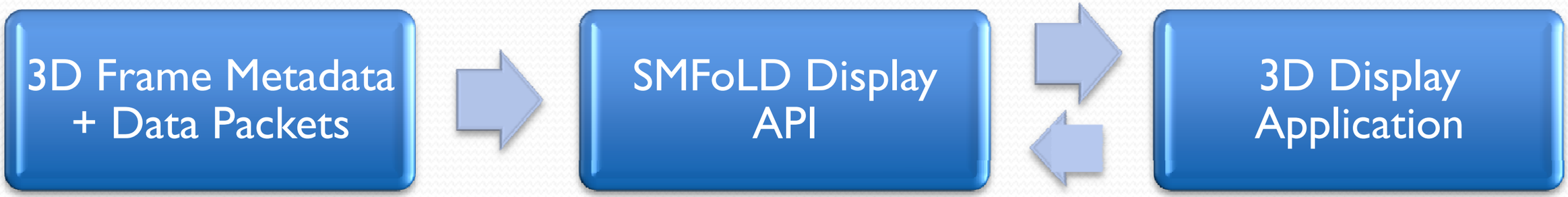
- Several Techniques Available
  - Triple DES
  - DSA
  - Blowfish
  - Twofish
  - AES
  - Source code to implement is readily available
- Key generation required
  - SMFoLD Source Process generates public/private keys
  - SMFoLD Display Process manages session key

# 3D Frame Decoded




- 3D Frames Decompressed and Decrypted
- 3D Data Restored to Metadata + Data Packets
- Potentially Cache Data Locally for Reuse
  - Model definitions
  - Shader code

# Display Creates GPU Code




- Display Driver Converts Data for Display Specific Rendering
  - Output can be determined by display (OpenGL, DirectX, other)
- Metadata Used to Setup Scene
- Textures
  - Could be flagged for reuse and stored on the display
- Shaders
  - Shader hooks used to set view geometry

# Metadata Extensions

- Metadata Provides Information to Display In-focus Views
- Geometry Limits (extents of the data)
  - Allow viewpoint changes to create multiple eye points
  - Information needed to create in-focus view with desired parallax
- Area of Interest (SMFoLD Source API Extension)
  - Defined either by Source or Display Application
  - Reduces the number of points used in calculations
  - Used to calculate metadata (average z) for the 3D Frame

# Shaders

- Graphics Pipeline is Implemented in Shaders
- Vertex Shader Performs Geometric Transformations
- Shader Hooks Allow Geometry Changes by Display Application
  - Hooks are named variables that display application can access (Viewpoint, Position, FOV, Data Extent . . .)
  - Source application required to provide hooks
- Shader Function in Header that Vertex Shader Required to Call
  - Function will use the named variables

# Technical Hurdles

- Large Volumes of Data
  - Average internet connection speed 15.3Mbps, peak 69.7Mbps (Akamai 2016)
  - Network may not support sufficient frame rates without significant compression
- Irregular Update Rates
  - Variations in Size of 3D Frames
  - Changes in Network Loads
- Synchronizing Audio
- Backward Compatibility
  - Pre-compiled shaders - no access to angle, position, FOV…

# Large Volumes of Data

## Potential Solutions

- Graphics Primitives Reduce Required Bandwidth
- Update Frames--Cache Reusable Data
  - Models downloaded once
- High Bandwidth Internet
- Compression
  - Several techniques available, new methods in development
  - New standards under development

# Irregular Frame Rates

## Potential Solutions

- No Problem at High Data Rate
- Audio Raises Synchronization Issues
  - Allow the audio to be ‘choppy’
  - Force frame rates to be consistent
    - Force to rate of slowest frame
    - Requires knowledge of entire frame set
- Reduce complexity of frames

# Audio

## Potential Solutions

- Audio Can Depend On Predictable Frame Rates
- Audio Duration Must Match Frame Render and Receive Rate
- Audio Packets Could be a Separate Stream
  - Use time stamps to keep streams synchronized
  - Simplifies decoding and parsing
- Testing Needed to Determine Best Approach

# Backwards Compatibility

## Potential Solutions

- Many Useful 3D Applications Exist
- Paradigm Familiar to Graphics Programmers
  - Uses OpenGL
  - SMFoLD DLL allows the process to be used as an interceptor
    - One way communications only
    - OpenGL 2.2 or earlier or applications that send shader source

# Phase II Tasks

- Develop Draft SMFoLD Standard Technical Report
- Conduct SMFoLD Workshops
- Implement SMFoLD Standard
  - Define packet structure of stream and SMFoLD extensions
  - Implement compression, encryption, audio, and change frame
  - Develop SMFoLD Source and Display Processes
  - Create and test Source and Display Applications
- Demonstrate on SMFoLD Compliant Display(s)
- Present Standard at Conferences
- Publish Reference Code

# Conclusions

- OpenGL API Graphics Primitives Plus Extensions Provides a Short Path to 3D Streaming
- High Frame Rates can be Achieved Over Existing Networks
- Different 3D Data Types can be Added
- Need Support of Application and Display System Providers

# Next Steps

- Feedback from Application and Display Providers
- Implement Model and Share Test Results
- Attend Next Workshop (TBD)
- Visit SMFoLD.org
- Final Technical Report (November, 2019)