**AI Material Texture Generation System Development Log**

*Note: Minor updates prior to major versions were implemented directly in source files without using a Git repository. The versions listed below represent major functional releases.*

**Version v1.0 - Base Architecture (tbeta.py)**

● ​**​Core Modules Implemented:​**​

1. Triple-File Scanner: Automatic OBJ/MTL/TGA file matching
2. Mesh Feature Extractor: Extraction of 33-dimensional geometric features (vertex statistics/curvature/normal vectors, etc.)
3. Data Loader: TGA texture loading support & basic data augmentation
4. Prediction Model: Dual-branch structure (MTL prediction + texture generation)
5. Training System: Custom losses (SSIM/VGG feature/gradient penalty)
6. Generation Pipeline: Automatic creation of complete material packages (OBJ+MTL+TGA)

● ​**​Key Technical Features:​**​

●● Residual connection network architecture

● Transposed convolution texture decoder

●● Perceptual loss based on VGG16

● Adaptive learning rate scheduling

**Version v1.1 - Model Architecture Optimization (t0.py)**

● ​**​Major Improvements:​**​

● √ Enhanced feature extraction: Added advanced features (curvature/normal vector extrema)

● √ Decoder refactoring: Streamlined upsampling path (4x4→256x256)

● √ Training stability:

* Gradient clipping (threshold=1.0)

● - Learning rate warm-up

* Early stopping (15-epoch patience)

√ Validation set: Random 10% data validation

● ​**​Issue Fixes:​**​

● x Fixed BatchNorm compatibility with small batches

● x Resolved NaN exceptions in feature tensors

x Optimized texture loading exception handling

**Version v1.2 - GAN Adversarial Training (tl.py)**

● ​**​Architectural Innovations:​**​

● ★ Introduced Generative Adversarial Network:

* Generator: UNet structure with skip connections
* Discriminator: 5-layer convolutional discriminator

● ★ Loss function upgrades:

* Adversarial loss (BCEWithLogits)
* Gram matrix style loss
* Gradient penalty term (WGAN-GP)

● ​**​Key Optimizations:​**​

●●● Enhanced data augmentation: Added rotations/flips & spatial transforms

●● Self-attention mechanism: Integrated attention modules in higher decoder layers

● Checkpoint saving: Optimizer state integration

● Training acceleration: Cosine annealing learning rate strategy

**Version v1.3 - File Processing Enhancement (t2.py)**

● ​**​OBJ/MTL Processing Upgrades:​**​

● Intelligent MTL matching: Prioritizes reading mtllibdeclarations in OBJs

● Material naming standardization: Unified {obj\_name}\_Materialformat

● OBJ rewriting logic:

* Automatic insertion of usemtldirectives
* Smart geometric data insertion point location
* Preservation of original model structure

● ​**​Error Handling Improvements:​**​

● √ Exception capture: Added traceback.print\_exc()for error tracing

● √ Fail-safe: Returns zero tensors on feature extraction failure

● √ Texture safety check: All-black texture alert system

√ Multi-encoding support: Handles special characters with errors='ignore'

**Version v1.4 - Engineering Reinforcement (t3.py)**

● ​**​Critical Stability Updates:​**​

● ​**​Parent Data Loading:​**​

● - Disabled PyTorch3D auto-texture loading (load\_textures=False)

· - Added OBJ parsing exception capture

● - Strengthened regex numerical matching ([ \d\e E+-]+$)

● ​**​Parent Memory Optimization:​**​

* Reduced point sampling (min(5000, len(verts)))

• - Cache clearance scheduling (torch.cuda.empty\_cache())

● ​**​Parent Logging Enhancements:​**​

• - Per-batch training metrics output

* Epoch time statistics
* Gradient norm monitoring

● ​**​Structural Optimizations:​**​

●● Efficient attention: Added spatial reduction (reduction=4)

●● Skip connections: Improved feature map alignment interpolation logic

● Decoder channel optimization: 192→128→96→64→48→32

**Development Version dev1.5 (t4.py)**

*Attempted implementation of partial training but unsuccessful.*

**System Evolution Key Metrics Comparison**

| **Version** | **Feature Dim** | **Training Strategy** | **Decoder Layers** | **Validation Method** | **Key Innovation** |
| --- | --- | --- | --- | --- | --- |
| v1.0 | 33 | Basic supervised | 12 layers | Random 10% sample | VGG perceptual loss |
| v1.1 | 33(+) | + Gradient clipping | 8 layers | Same | Learning rate warm-up |
| v1.2 | 33 | GAN adversarial | 12L + attention | Same | WGAN-GP framework |
| v1.3 | 33 | Same | Same | Same | Smart OBJ rewriting engine |
| v1.4 | 33 | + Memory optim | Optimized channels | Same | Efficient attention module |

**Cumulative Issues Resolved**

1. Texture size mismatch (v1.1)
2. Small-batch training crash (v1.1)
3. MTL-Kd parsing exception (v1.3)
4. Memory leakage (v1.4)
5. Feature NaN contamination (all versions)
6. OBJ material reference breakage (v1.3)

*This development log comprehensively documents the system's evolution from foundational version to production-grade solution, including key technical decisions, issue resolution paths, and architectural optimization milestones. All content is in plain text format for direct use in Word documents.*