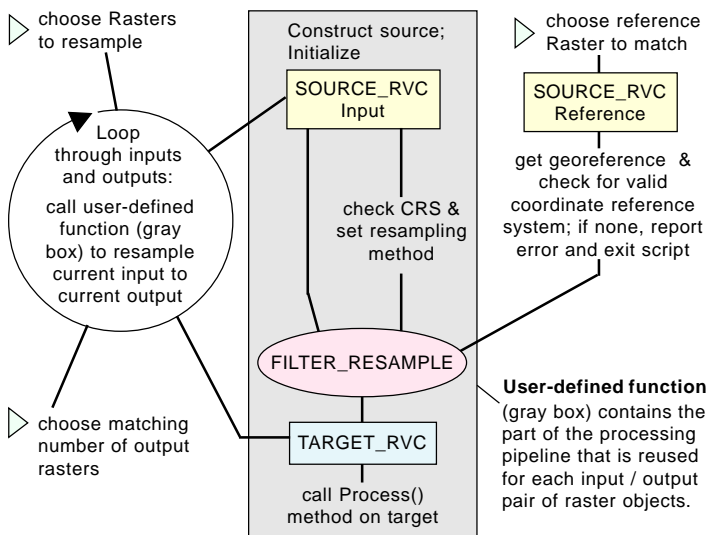# Pipeline Structures for Multiple Inputs

A previous Technical Guide (*Geospatial Scripting: Pipeline Programming Basics*) presents an example of a relatively simple pipeline programming task using the TNT Geospatial Scripting Language (SML): resampling a raster object from a MicroImages Project File to match a reference raster object. But what if you want to resample any number of raster objects (for example, six bands of a Landsat scene, or a color composite and an elevation raster) to match that same reference image? In effect, you want to reuse the same image processing pipeline for each of the input rasters to create a unique output raster. The solution is to wrap the "reuseable" portion of the pipeline within a user-defined procedure or function within the script. The illustration below diagrams the design of such a script, PipelineResampleToMatchMulti.sml, which is also excerpted on the reverse.
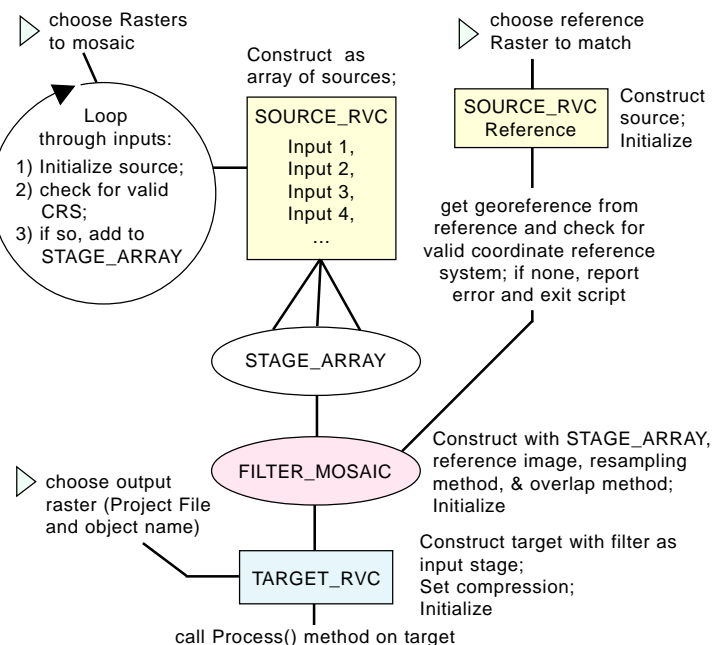

Annotated diagram of pipeline script to resample multiple images to match a reference image (see excerpts of PipelineResampleToMatchMulti.sml on reverse).

Each of the raster objects you select as input for such a script must be set up as a unique image source paired with an associated image target. The sample script thus requires matching lists of RVC_OBJITEM class instances for the sources and targets, and the number of items is not known until the input raster objects are selected. Such "lists" can be set up by declaring an instance of class RVC_OBJITEM as a *hash* (a data class similar to an array). The input hash is populated using a predefined function [DlgGetObjects() in the Popup Dialog function group] that provides a dialog to prompt the user to select any number of input raster objects and reads their RVC_OBJITEMS into the specified hash. A second function [DlgGetObjectSet()] to select a specified number of objects (set to be the same as the number of inputs selected) is used to populate the hash of output RVC_OBJITEMS.

The script contains a processing loop that iterates through the hashes of input and output RVC_OBJITEMS and passes each pair to the user-defined function, where they are used to construct the source and target stages for that iteration. Since the same reference image is used for each pass, the reference source is constructed and initialized only once, outside the user-defined function, and its class instance is also passed to the function for use in constructing the resampling filter stage for the current iteration. Within the user-defined function, the input source, resampling filter, and target stage are all constructed and initialized and the Process() method is called for the target to process the current image through the pipeline.

A pipeline script to create a mosaic from any number of input images, such as the example diagrammed below and excerpted on the reverse (PipelineMosaicToReference.sml), requires a somewhat different script structure. The Mosaic filter uses an instance of class IMAGE_PIPELINE_STAGE_ARRAY to hold all of the images to be mosaicked. Each of the unique sources that populate the STAGE_ARRAY must remain valid when the pipeline is processed, so the script can't reuse the same SOURCE_RVC class instance to populate the STAGE_ARRAY. You also don't know how many sources the script will need until the input images are selected. The solution is to construct the SOURCE_RVC class as an array of class instances. The script uses a processing loop to add sources (indexed by number) to the array of sources using a corresponding hash of RVC_OBJITEMS from the selected input raster objects. This loop also get the georeference information from each input and determines if it has a valid, non-local coordinate reference system (CRS; it can be different from that of other inputs or the reference image); if so, that source is appended to the stage array, whereas any input raster lacking a valid CRS is simply skipped. The mosaic filter is then constructed using the STAGE_ARRAY and a reference source that sets the extents, CRS, and cell size of the mosaic.

Annotated diagram of pipeline script to mosaic multiple images to match a reference image (see excerpts of PipelineMosaicToReference.sml on reverse).

Excerpts of Pipeline Script to Mosaic Images to Match a Reference Image
(PipelineMosaicToReference.sml)

[code to set up and initialize source for reference image omitted]

CHOOSE GEOREFERENCED RASTERS TO MOSAIC

HASH of RVC_OBJITEMS for unknown number of input rasters to mosaic

```
class RVC_OBJITEM inObjItemList[];
numeric numInputs;
```

DlgGetObjects populates an RVC_OBJITEM hash with the RVC_OBJITEM of each selected raster

```
DlgGetObjects("Choose georeferenced rasters to mosaic:", "Raster", inObjItemList,
"ExistingOnly", 2);
```

set up array of RVC_SOURCE class handles with number equal to number of input rasters

```
numInputs = inObjItemList.GetNumItems();
class IMAGE_PIPELINE_SOURCE_RVC sources[numInputs];
printf("Number of rasters to mosaic = %d\n", numInputs);
```

set up PIPELINE STAGE_ARRAY to pass to the mosaic filter; construct with unspecified number of stages to allow for skipping ungeoreferenced inputs

```
class IMAGE_PIPELINE_STAGE_ARRAY stages( );
```

declare class variables to use in processing the input rasters

```
class RVC_OBJITEM objItem;
class IMAGE_PIPELINE_GEOREFERENCE sourceGeoref;
class SR_COORDREFSYS crs;
```

loop through the hash of input objItems

get ObjItem for current source image from hash of RVC_OBJITEMS

```
for i = 1 to numInputs {
    objItem = inObjItemList[i];
```

PIPELINE SOURCE for each input image; add a new source for the current image to the source array and initialize

```
    sources[i] = new IMAGE_PIPELINE_SOURCE_RVC(objItem);
    err = sources[i].Initialize();
    if ( err < 0 ) ReportError(_context.CurrentLineNum, err);
    else  printf("\nInitialized source %d of %d\n", i, numInputs);
```

check that source has valid coordinate reference system; if valid, add source to STAGE_ARRAY, otherwise skip

```
    sourceGeoref = sources[i].GetGeoreference();
    crs = sourceGeoref.GetCRS();

    if (crs.IsDefined() == 0 or crs.IsLocal() ) {
        printf("Coordinate reference system
                    for source %d is undefined or local.\n",  i);
        printf("Source %d will be omitted from mosaic.\n", i);
        }
    else {
        printf("Source %d: CRS = %s\n", i, crs.Name );
        stages.Append(sources[i]);
        }
    }
```

append source to the STAGE_ARRAY

CHOOSE OUTPUT RASTER

ObjItem for the output mosaic raster

```
class RVC_OBJITEM mosObjItem;
DlgGetObject("Select new raster object for the mosaic:", "Raster", mosObjItem,
    "NewOnly");
```

PIPELINE FILTER to mosaic the input rasters

```
class IMAGE_PIPELINE_FILTER_MOSAIC mosaic(stages, refSource,
    "Nearest", "Last");
err = mosaic.Initialize();
if ( err < 0 ) ReportError(_context.CurrentLineNum, err);
else print("Mosaic filter initialized.");
```

PIPELINE TARGET: set up the target for the mosaic pipeline

```
class IMAGE_PIPELINE_TARGET_RVC target_rvc(mosaic, mosObjItem);
target_rvc.SetCompression("DPCM", 0);
err = target_rvc.Initialize();
if (err < 0)  ReportError(_context.CurrentLineNum, err);
else  print("Pipeline target initialized.");

print("Processing...");
target_rvc.Process();
```

EXECUTE pipeline process

```
print("Done.");
```

Excerpts of Pipeline Script to Resample Multiple
Images to Match a Reference Image
(PipelineResampleToMatchMulti.sml)

[code to set up and initialize source for reference image omitted]

CHOOSE GEOREFERENCED INPUT RASTERS to be resampled.

HASH of RVC_OBJITEMS for unknown number of input rasters to resample

```
class RVC_OBJITEM inObjItemList[];
numeric numInputs;
```

DlgGetObjects populates an RVC_OBJITEM hash with the RVC_OBJITEM of each selected raster

```
DlgGetObjects("Choose georeferenced rasters to resample:", "Raster",
                inObjItemList, "ExistingOnly", 2);
numInputs = inObjItemList.GetNumItems();
printf("Processing %d input rasters.\n\n", numInputs);
```

MAKE STRINGLIST OF LABELS FOR DIALOG PROMPTING FOR OUTPUT RASTERS

```
class STRINGLIST labelList;
for i = 1 to numInputs
    {
    labelList.AddToEnd( inObjItemList[i].GetDescriptor().GetShortName() );
    }
```

CHOOSE OUTPUT RASTERS

DlgGetObjectSet populates an RVC_OBJITEM hash with the RVC_OBJITEM of each selected raster

```
class RVC_OBJITEM outObjItemList[];
DlgGetObjectSet("Choose rasters for resampled output", "Raster", labelList,
                outObjItemList, "NewOnly");
```

LOOP THROUGH HASHS OF INPUT AND OUTPUT RVC_OBJITEMS TO RESAMPLE IMAGES

```
for i = 1 to numInputs
    {
```

CALL USER-DEFINED FUNCTION TO RESAMPLE AN IMAGE

```
    err = rsmpPipe(inObjItemList[i], outObjItemList[i], source_Ref, refCellArea,
            i, labelList);
```

[function definition omitted]

invalid coordinate reference system for input

```
    if (err == 0)
        printf("Source image %d has undefined or local coordinater reference
                system; \n no resampled raster was made.\n\n", i);
    else if (err < 0)      ReportError(_context.CurrentLineNum, err);
    }

print("Done.");
```