EFFECTIVENESS OF EXTRACTING WATER SURFACE SLOPES FROM LIDAR DATA WITHIN THE ACTIVE CHANNEL: SANDY RIVER, OREGON, USA

by

JOHN THOMAS ENGLISH

A THESIS

Presented to the Department of Geography
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

March 2009

"Effectiveness of Extracting Water Surface Slopes from LiDAR Data within the Active Channel: Sandy River, Oregon, USA," a thesis prepared by John Thomas English in partial fulfillment of the requirements for the Master of Science degree in the Department of Geography. This thesis has been approved and accepted by:

W. Andrew Marcus, Chair of the Examining Committee

2/25/2009
Date

Committee in Charge: W. Andrew Marcus, Chair
Patricia F. McDowell

Accepted by:

Dean of the Graduate School

An Abstract of the Thesis of

John Thomas English for the degree of Master of Science

in the Department of Geography to be taken March 2009

Title: EFFECTIVENESS OF EXTRACTING WATER SURFACE SLOPES FROM LIDAR DATA WITHIN THE ACTIVE CHANNEL: SANDY RIVER, OREGON, USA

Approved: ______________________________________

W. Andrew Marcus

This paper examines the capability of LiDAR data to accurately map river water surface slopes in three reaches of the Sandy River, Oregon, USA. LiDAR data were compared with field measurements to evaluate accuracies and determine how water surface roughness and point density affect LiDAR measurements. Results show that LiDAR derived water surface slopes were accurate to within 0.0047, 0.0025, and 0.0014 slope, with adjusted R^2 values of 0.35, 0.47, and 0.76 for horizontal intervals of 5, 10, and 20m, respectively. Additionally, results show LiDAR provides greater data density where water surfaces are broken. This study provides conclusive evidence supporting use of LiDAR to measure water surface slopes of channels with accuracies similar to field based approaches.

CURRICULUM VITAE

NAME OF AUTHOR: John Thomas English

PLACE OF BIRTH: Eugene, Oregon

DATE OF BIRTH: January 1st, 1980

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, Oregon
Southern Oregon University, Ashland, Oregon

DEGREES AWARDED:

Master of Science, Geography, March 2009, University of Oregon
Bachelor of Science, Geography, 2001, Southern Oregon University

AREAS OF SPECIAL INTEREST:

Fluvial Geomorphology
Remote Sensing

PROFESSIONAL EXPERIENCE:

LiDAR Database Coordinator, Oregon Department of Geology & Mineral Industries, June 2008 - present.

LiDAR & Remote Sensing Specialist, Sky Research Inc., 2003 - 2008

GRANTS, AWARDS AND HONORS:

Gamma Theta Upsilon Geographic Society Member, 2006

Gradutate Teaching Fellowship, Social Science Instructional Laboratory, 2006-2007

ACKNOWLEDGMENTS

I wish to express special thanks to Professors W.A. Marcus and Patricia McDowell for their assistance in the preparation of this manuscript. In addition, special thanks are due to Mr. Paul Blanton who assisted with field data collection for this project. I also thank the members of my family who have been encouraging and supportive during the entirety of my graduate schooling. I wish to thank my parents Thomas and Nancy English for always being proud of me. Special thanks to my son Finn for always making me smile. Lastly, special thanks to my wife Kathryn for her unwavering support, love, and encouragement.

Dedicated to my mother Bonita Claire English (1950-2004).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table | Page

CHAPTER I

INTRODUCTION

LiDAR (Light Detection and Ranging) has become a common tool for mapping and documenting floodplain environments by supplying individual point elevations and accurate Digital Terrain Models (DTM) (Bowen & Waltermire, 2002; Gilvear et al., 2004; Glenn et al., 2005; Magirl et al., 2005; Thoma, 2005; Smith et al., 2006; Gangodagamage et al., 2007). Active channel characteristics that have been extracted using LiDAR include bank profiles, longitudinal profiles (Magirl et al., 2005; Cavalli et al., 2007) and transverse profiles of gullies under forest canopies (James et al., 2007). To date, however, no one has tested if LiDAR returns from water surfaces can be used to measure local water surface slopes within the active channel.

Much of the reason that researchers have not attempted to measure water surface slopes with LiDAR is because most LiDAR pulses are absorbed or not returned from the water surface. However, where the angle of incidence is close to nadir (i.e. the LiDAR pulse is fired near perpendicular to water surface plane), light is reflected and provides elevations off the water surface (Figure 1, Maslov et al., 2000). Where LiDAR pulses glance the water surface at angles of incidence greater than 53 degrees, a LiDAR pulse is more often lost to refraction (Figure 2) (Jenkins, 1957). In broken water surface conditions the water surface plane is angled, which produces perpendicular angles of incidence allowing for greater chance of return (Maslov et al. 2000). Su et al. (2007) documented this concept by examining LiDAR returns off disturbed surfaces in a controlled lab setting (Figure 3). LiDAR returns off the water surface potentially provide accurate surface elevations that can be used to calculate surface slopes.

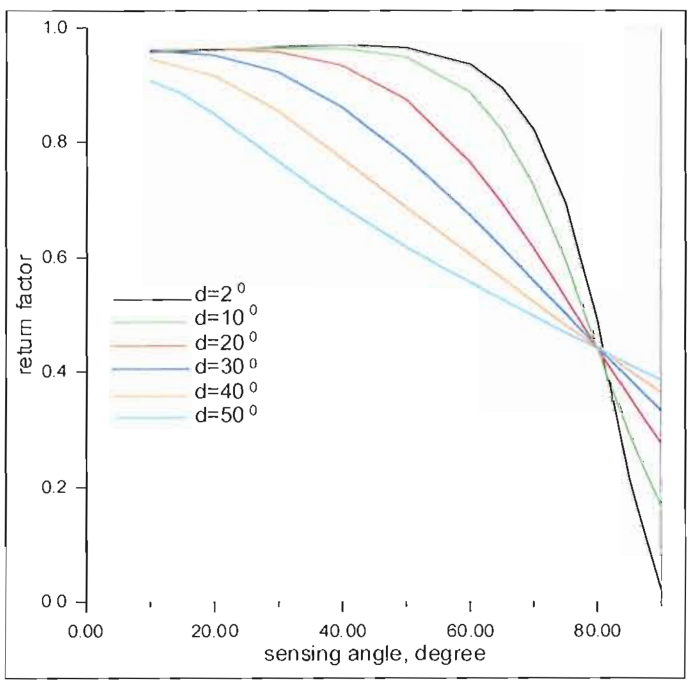


Figure 1. Return Factor vs. LiDAR Scan Angle. Figure shows relationship between water surface return and scan angle. Return Factor versus sensing angle at different levels of the waving d (d = scan angle). Figure shows the relationship of scan angle of LiDAR to return from a water surface. Return factor is greatest at low scan angles relative to the nadir region of scan. (Maslov, D. V. et. al. (2000). A Shore-based LiDAR for Coastal Seawater Monitoring. Proceedings of EARSeL-SIG-Workshop, Figure 1, pg. 47).

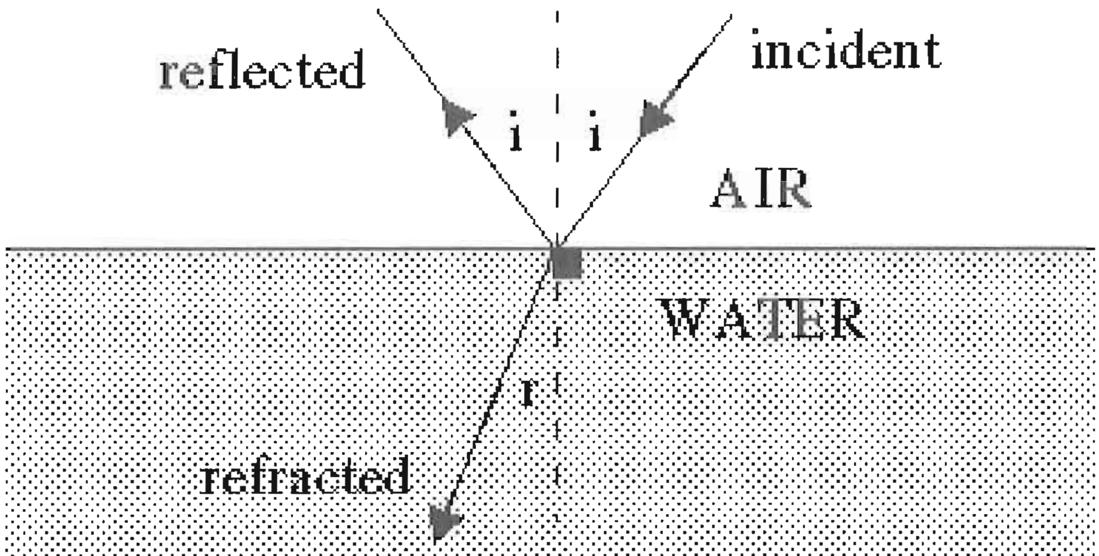
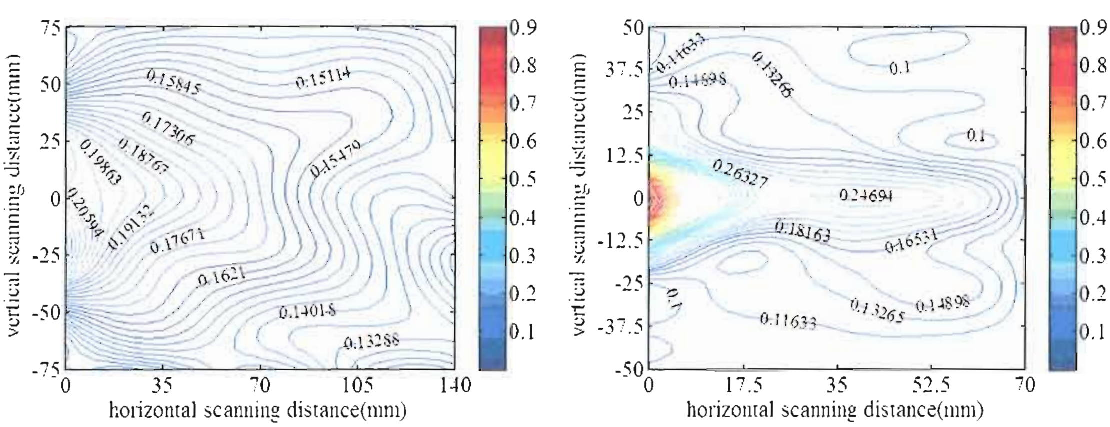


Figure 2. Angle of Incidence. Figure displays concept of reflection and refraction of light according to angle of incidence. The intensity of light is greater as the angle of incidence approaches nadir. (Jenkins, F.A., White, H.E. "Fundamentals of Optics". McGraw-Hill, 1957, Chapter 25)



a b

Figure 3. Wave Action Relationship to LiDAR Echo. "LiDAR measurements of wake profiles generated by propeller at 6000 rpm (a) and 8000 rpm (b). Su's work definitively showed LiDAR's ability to measure water surfaces, and the relationship of wave action to capability of echo. From Su (2007) figure 5, p.844 .

This study examines whether LiDAR can accurately measure water surface elevations and slopes. In order to address this topic, I assess the vertical accuracy of LiDAR and the effects of water surface roughness on LiDAR within the active channel. Findings shed light on the utility of LiDAR for measuring water surface slopes in different stream environments and methodological constraints to using LiDAR for this purpose.

CHAPTER II

BACKGROUND

Water Surface Slope

Water surface slope is a significant component to many equations for modeling hydraulics, sediment transport, and fluvial geomorphic processes (Knighton, 1999, Sing & Zang, in press). Traditional methods for measuring water surface slope include both direct and indirect methods. Direct water surface slope measurements typically use a device such as a total station or theodolite in combination with a stadia rod or drop line to measure water surface elevations (Harrelson, et al., 1994, Western et al., 1997). Inaccuracies in measurements stem from surface turbulence that makes it difficult to precisely locate the water surface, especially in fast water where flows pile up against the measuring device (Halwas, 2002). Direct survey methods often require a field team to occupy several known points throughout a reach. This is a time consuming process, especially if one wanted to document water surface slope along large portions of a river. This method can be dangerous in deep or fast water.

Indirect methods of water surface slope measurement consist of acquiring approximate water surface elevations using strand lines, water marks, secondary data sources such as contours from topographic maps, or hydraulic modeling to back calculate the water depth (USACE, 1993; Western et al., 1997). Variable quality of data and modeling errors can lead to inaccuracies using these methods. The use of strand lines and water marks may not necessarily represent the peak flows or the water surface. Contours may be calculated or interpolated from survey points taken outside the channel area. The most commonly used hydraulic models are based on reconstruction of 1-dimensional flow within the channel and do not account for channel variability between cross section locations.

LiDAR water surface returns have a great deal of promise for improving measurement of water surfaces in several significant ways. LiDAR measurements eliminate hazards associated with surveyors being in the water. LiDAR also captures an immense amount of elevation data over a very short period of time, with hundreds of thousands of pulses collected within a few seconds for a single swath. Within this mass of pulses, hundreds or thousands of measurements off the water's surface may be collected depending on the nature of surface roughness, with broken water surfaces increasing the likelihood of measurements (Figure 3). In addition, most terrestrial LiDAR surveys collect data by flying multiple overlapping flight lines, thus increasing the number of returns in off nadir overlapping areas and the potential for returns from water surfaces.

The accuracy of high quality LiDAR measurements is comparable to field techniques. The relative variability of quality LiDAR vertical measurements typically ranges between 0.03-0.05 meters (Leica, 2007), where relative variability is the total range of vertical error within an individual scan on surface of consistent elevation. Lastly, LiDAR has the ability to collect water surface elevations over large stretches of river within a single flight of a few hours.

LiDAR Measurements of Active Channel Features

Recent studies evaluating the utility of LiDAR in the active channel environment have documented the effectiveness of using LiDAR DTMs to extract bank profiles. Magirl et al. (2005) examined long term changes of longitudinal profiles along the Colorado River in the Grand Canyon. The study used historical survey data from 1923 and differenced topographic elevations with LiDAR data flown in 2000. LiDAR with three meter spot spacing was used to estimate water surface profiles based on the LiDAR elevations nearest to the known channel. Cavalli et al. (2007) extracted longitudinal profiles of the exposed bed of the Rio Cordon, Italy using 0.5 meter LiDAR DEM cells. This study successfully attributed LiDAR DEM roughness within the channel to in-stream habitats. Bowen and Waltermire (2002) found that LiDAR elevations within the floodplain were less accurate than advertised by vendors and sensor manufacturers. Dense vegetation within the riparian area prevented LiDAR pulses from reaching the ground surface resulting in accuracies ranging 1-2 meters. Accuracies within unvegetated areas and flat surfaces met vendor specifications (15-20cm).

James et al. (2007) used LiDAR at 3 meter spot spacing to map transverse profiles of gullies under forest canopies. Results from this study showed that gully morphologies were underestimated by LiDAR data, possibly due to low density point spacing and biased filtering of the bare earth model. Today, point densities of 4-8 points/m^2 are common and would likely alleviate some of the troubles found in this study.

Additional studies have used LiDAR to extract geomorphic data from channel areas. Schumann et al. (2008) compared a variety of remotely sensed elevation models for floodplain mapping. The study used 2 meter LiDAR DEMs as topographic base data for floodplain modeling, and found that modeled flood stages based on the LiDAR DEM were accurate to within 0.35m. Ruesser and Bierman (2007) used high resolution LiDAR data to calculate erosion fluxes between strath terraces based on elevation. Gangodagamage et al. (2007) used LiDAR to extract river corridor width series, which help to quantify processes involved in valley formation. This study used a fixed water surface elevation and did not attempt to demonstrate the accuracy of LiDAR derived water surfaces.

Green LiDAR also has been used to examine riverine environments. Green LiDAR functions much like terrestrial LiDAR (which uses an infrared laser) except that green LiDAR systems use green light that has the ability to penetrate the water surface and measure the elevation of the channel bed. Green LiDAR is far less common than terrestrial LiDAR and the majority of studies have been centered on studies of ocean shorelines. Wang and Philpot (2007) assessed attenuation parameters for measuring bathymetry in near shore shallow water, concluding that quality bathymetric models can be achieved through a number of post-processing steps. Hilldale and Raft (2007) assessed the accuracy and precision of bathymetric LiDAR and concluded that although the resulting models were informative, bathymetric LiDAR was less precise than traditional survey methods. In general, it is often difficult to assess the accuracy of bathymetric LiDAR given issues related to access of the channel bed at time of flight.

CHAPTER III

STUDY AREA

The study area is the Sandy River, Oregon, which flows from the western slopes of Mount Hood northwest to the Columbia River (Figure 4). Recent LiDAR data and aerial photography capture the variety of water surface characteristics in the Sandy River, which range from shooting flow to wide pool-riffle formations. The recent removal of the large run-of-river Marmot Dam upstream of the analysis sites has also generated interest in the river's hydraulics and geomorphology.

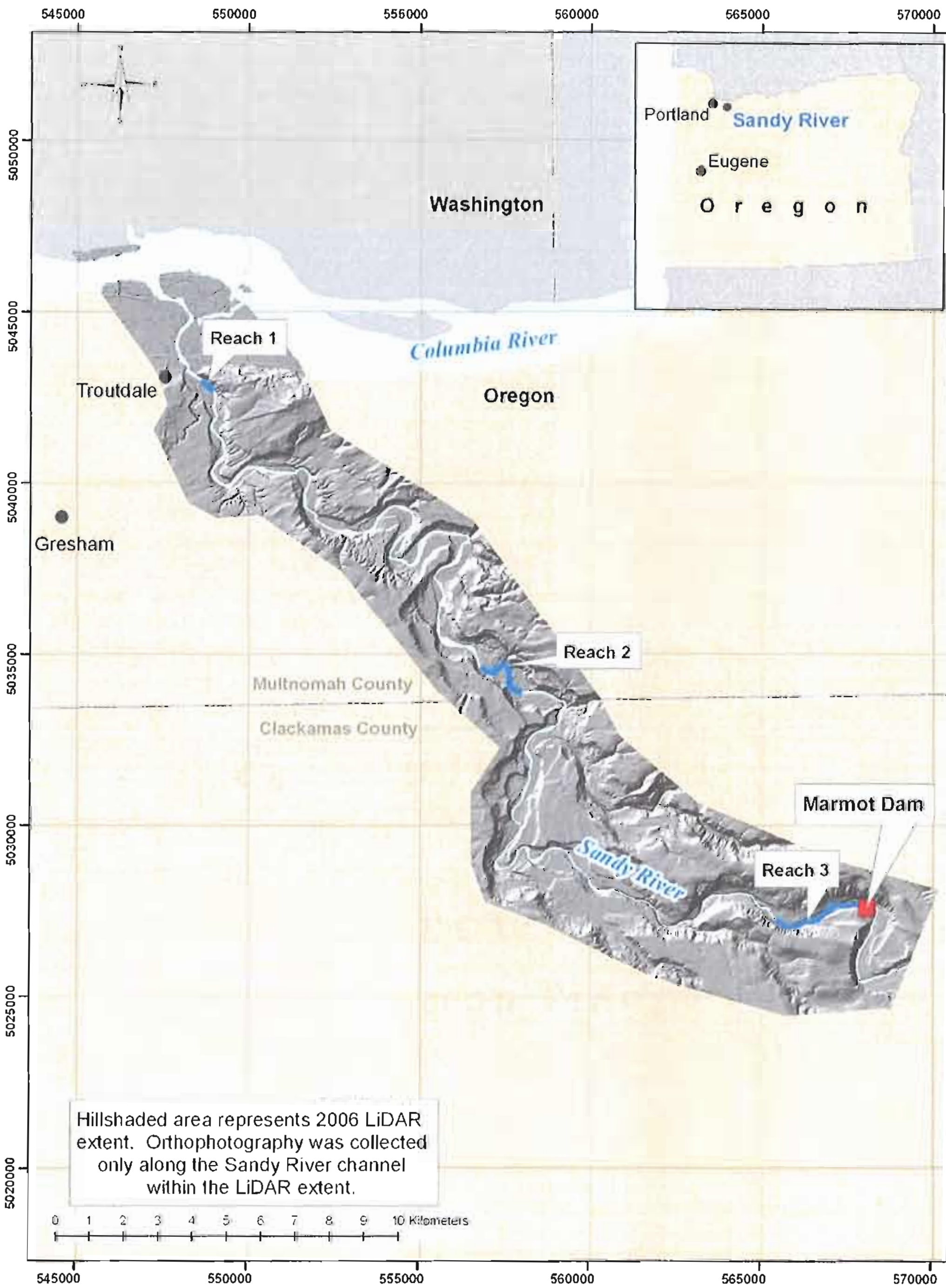


Figure 4. **Site Map.** Site area map showing location of analysis reaches within the 2006 and 2007 LiDAR coverage areas. Orthophotography was also collected for the 2006 study, but was collected only along the Sandy River channel.

Floodplain longitudinal slopes along the Sandy River average 0.02 and reach a maximum of 0.04. The Sandy River has closely spaced pool-riffles and rapids in the upper reaches, transitioning to longer sequenced pool-riffle morphology in the middle and lower reaches. The Sandy River bed is dominated by sand. Cobbles and small boulders are present mostly in areas of riffles and rapids. Much of the channel is incised with steep slopes along the channel boundaries. The flow regime is typical of Pacific Northwest streams, with peak flows in the winter months of November through February and in late spring with snowmelt runoff (Figure 5). Low flows occur between late September and early October. The average peak annual flow at the Sandy River station below Bull Run River (USGS 14142500) is 106cms. Average annual low flow for the same gauge is 13.9cms.

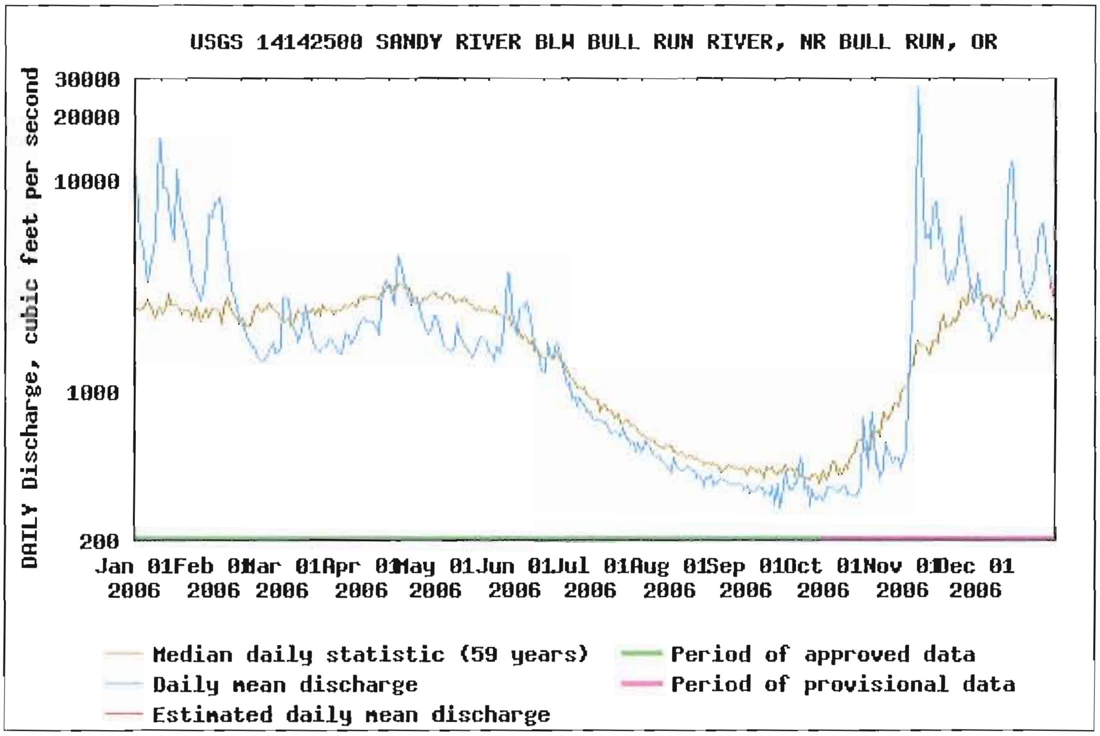


Figure 5. Annual Hydrograph of Sandy River. US Geological Survey gaging station annual hydrograph of Sandy River, Oregon at Bull Run River. Data from http://waterdata.usgs.gov/or/nwis/annual/

Vegetation is mostly a mixture of Douglas fir and western red hemlock (Figure 6). Other vegetation includes palustrine forest found in the upper portions of the study area, and agricultural lands found in the middle and lower portions. Douglas fir and western red hemlock make up 87% of vegetated areas, palustrine forest 5%, and agricultural lands 5%, the remaining 3% is open water associated with the channel and reservoirs (Oregon GAP Analysis Program, 2002). The city of Troutdale, OR abuts the lower reaches of the Sandy River. Along this stretch of river Himalayan blackberry, an invasive species, dominates the western banks (Figure 7). The presence of Himalayan blackberry is significant because LiDAR has trouble penetrating through the dense clusters of vines. When this blackberry is close to the water's edge it is difficult to accurately define the channel boundary.

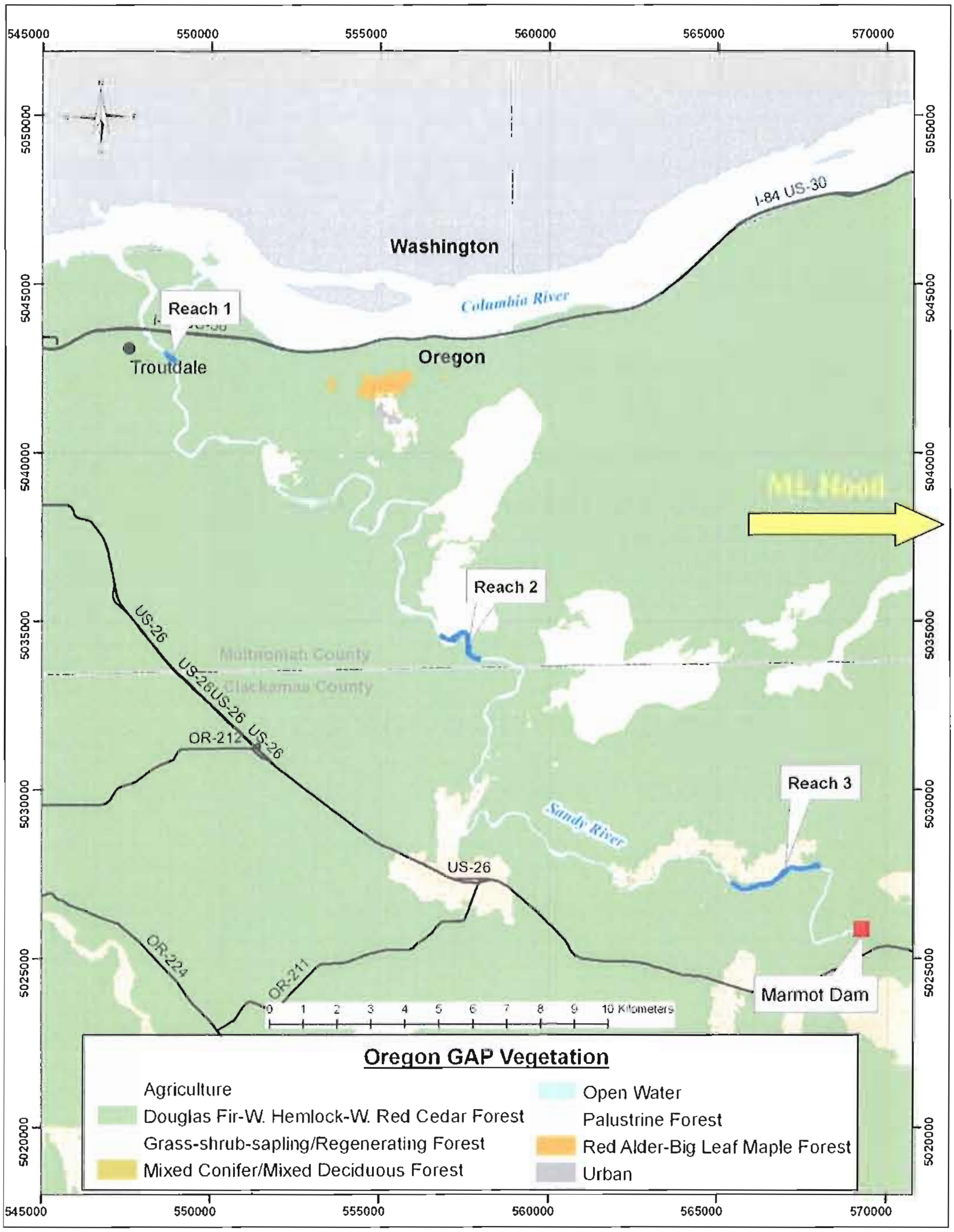
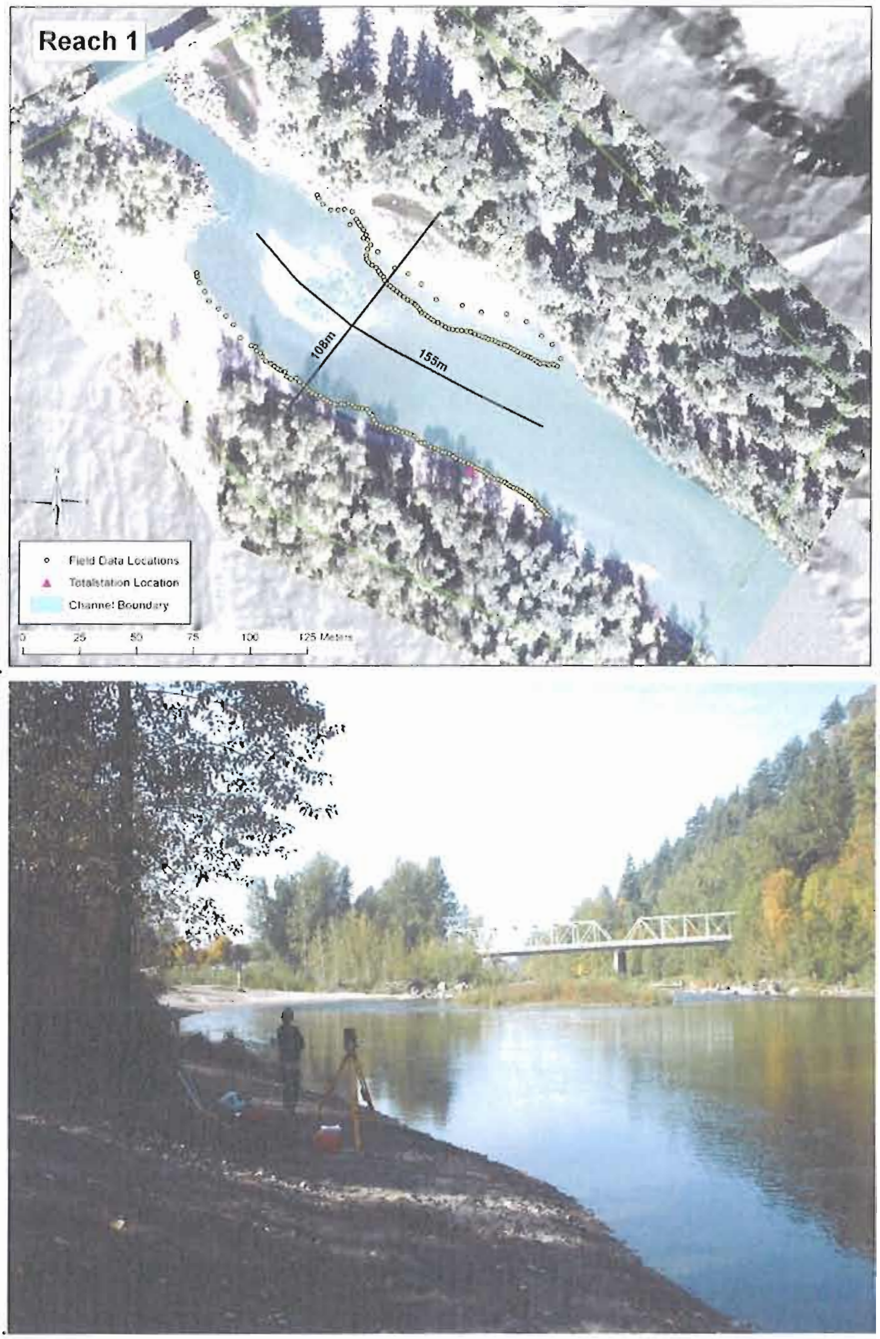


Figure 6. Oregon GAP Vegetation within Study Area. 1999 Oregon GAP Analysis data for Sandy River area. Map shows how the Sandy River area is dominated by Douglas fir forest with areas of palustrine forest and agricultural lands (Oregon Natural Heritage Program, 1999).

Figure 7. Photo of Himalayan Blackberry on Sandy River. Himalayan blackberry near mouth of the Sandy River March, 25th 2007. Photo by John English.

This study focuses on three reaches of channel that represent a range of water surface conditions along the river. Reach 1 is a 180-m long pool-riffle reach located 3.7 river kilometers upstream from the mouth, and is where we collected field data shortly after the 2007 LiDAR flight (Figure 8a). The bed is sandy in this reach and can change dramatically during high flows. The bank full width of Reach 1 is approximately 108 meters at its widest point. At the downstream end of the riffle, the channel is constricted by riprap placed along the banks as the river flows under a bridge. Vegetation comprises deciduous and conifer trees such as Douglas fir, hemlock, and cottonwoods. Blackberry is present along the channel, but is not so dense that it obscures the active channel boundary.



a.
b.

Figure 8. **Reach 1 Site Area Map with Photo.** Reach 1 site area. Top figure (a) shows approximate width at bank full and length of field data collections. Yellow circles represent points along stream margins where water surface elevations were surveyed. Bottom photo (b) looks downstream from total station location.

Reach 2 (Figure 9) is located approximately 23.5 km upstream from the mouth of the Sandy River and is 1,815 meters in length. The widest portion of channel at approximate bank full is 116m. The channel consists of a large meander with sinuosity of 1.38 and consists of six riffles and five pools spaced at regular intervals. The substrate consists of sands with small boulders and large cobbles dominating riffle areas. Cobbles and boulders have likely been introduced to the channel as a result of mass wasting. Douglas fir dominates along banks.

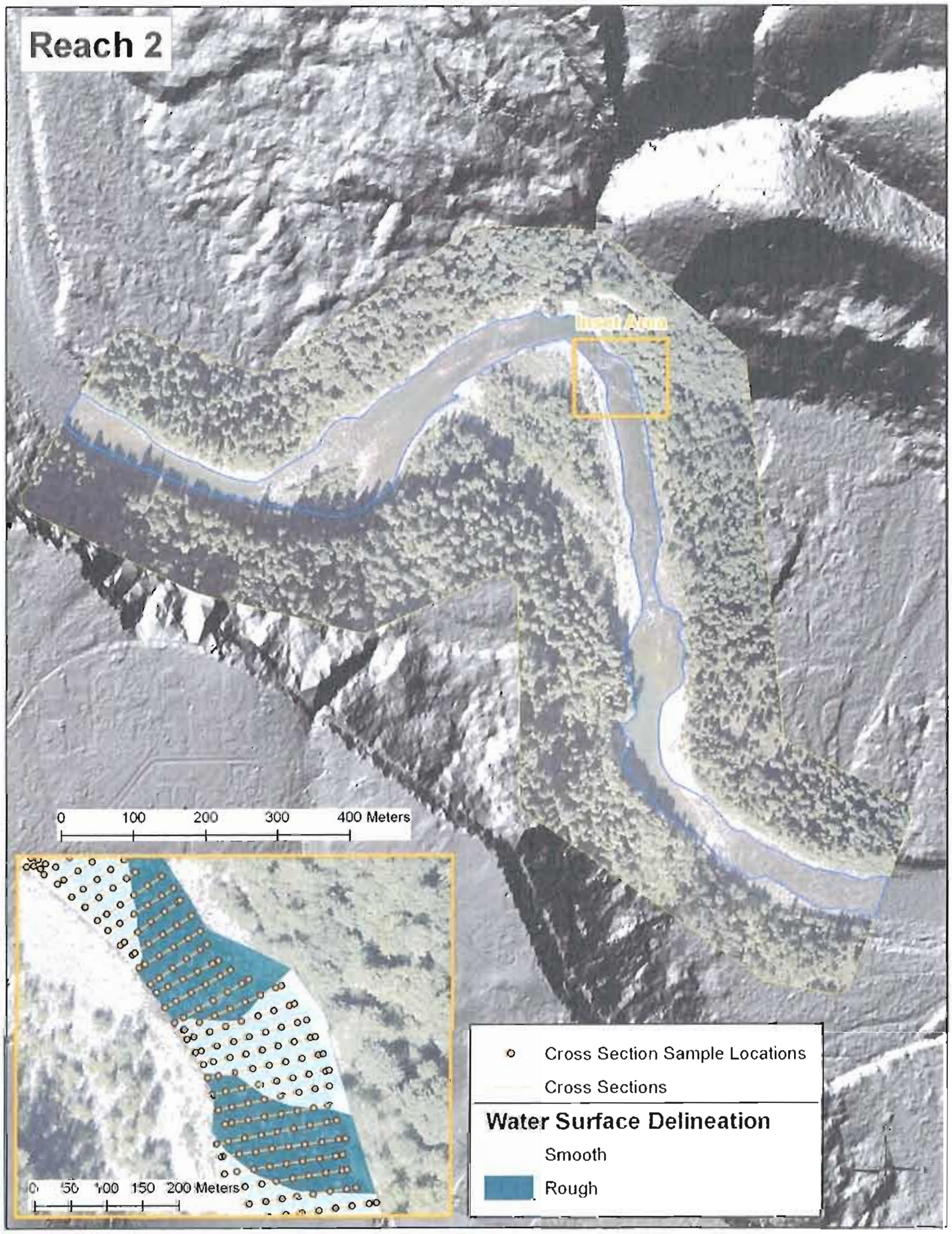


Figure 9. Reach 2 Site Area Map. Site map of Reach 2. Reach 2 contains 359 cross sections derived from LiDAR and 3,456 sample points. Inset map shows cross section sample locations derived from LiDAR and smooth/rough water surface delineations used in analysis.

Reach 3 is located 40.7km upstream from the mouth of the Sandy and is 2,815 meters in length (Figure 10). The widest portion of this section at approximate bank full is 88 meters. The upstream extent of the channel includes the supercritical flow of Marmot Dam. The channel is incised and relatively straight with a sinuosity of 1.08. Fine sands dominate the channel bed with some boulders likely present from mass wasting along valley walls. As with Reach 2, Douglas fir dominates bank vegetation along.

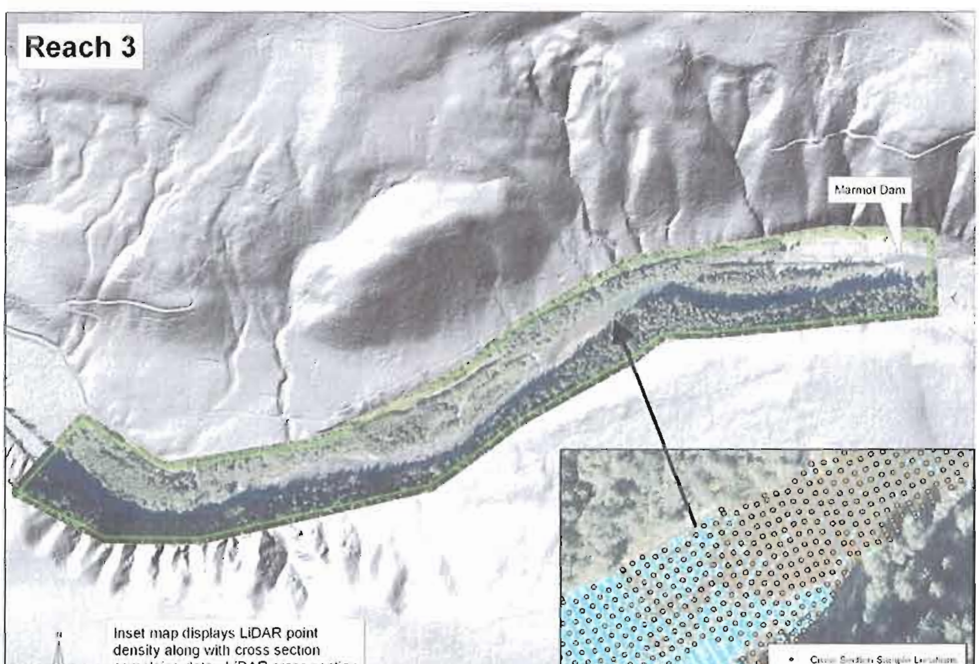


Figure 10. Reach 3 Site Area Map. Site map of Reach 3. Inset map shows point LiDAR water surface points. Reach 3 contains 550 cross sections and 3,348 sample points. Visual examination of this map allows one to see how point density varies within the active channel.

CHAPTER IV

METHODS

Overview

LiDAR data and orthophotography were collected in 2006 and additional LiDAR data were collected over the same area in 2007. Field measurements were obtained five days after the 2007 LiDAR flight in order to compare field measurements of water surface slope to LiDAR-based measurements. Time of flight field measurements of water surface elevations were not obtained for the 2006 flight, but the coincident collection of LiDAR data and orthophotos provide a basis for evaluating variability of LiDAR-based slopes over different channel types as identified from aerial photos. Following sections provide more detail regarding these methods.

LiDAR Data and Image Acquisition

All LiDAR data were collected using a Leica ALS50 Phase II LiDAR system mounted on a Cessna Caravan C208 (see Table 1 for LiDAR acquisition specifications). The 2006 LiDAR data were collected October 22nd and encompassed 13,780 hectares of high resolution (≥4 points/m^2) LiDAR data from the mouth of the Sandy River to Marmot Dam. Fifteen centimeter ground resolution orthophotography was collected September 26th, 2006 along the riparian corridor of the Sandy River from its mouth to just above the former site of Marmot dam (Figure 4). The 2007 LiDAR were collected on October 8th and covered the same extent as the 2006 flight, but did not include orthophotography. Data included filtered XYZ ASCII point data, LiDAR DEMs as ESRI formatted grids at 0.5 meter cell size. Data were collected at ≥8 points per m^2 providing a data set with significantly higher point density than the 2006 LiDAR data.

The 2006 LiDAR data were collected in one continuous flight. 2006 orthophotography was collected using an RC30 camera system. Data were delivered in RGB geoTIFF format. LiDAR data were calibrated by the contractor to correct for IMU position errors (pitch, roll, heading, and mirror scale). Quality control points were collected along roads and other permanent flat features for absolute vertical correction of data. Horizontal accuracy of LiDAR data is governed by flying height above ground with horizontal accuracy being equal to 1/3300th of flight altitude (meters) (Leica, 2007).

Table 1. Reported Accuracies of 2006 and 2007 LiDAR. Reported Accuracies and conditions for 2006 and 2007 LiDAR data. (Watershed Sciences PGE LiDAR Delivery Report, 2006, Watershed Sciences DOGAMI LiDAR Delivery Report, 2007). Relative Accuracy is a measure of flight line offsets resulting from sensor calibration.

	2006 LiDAR	**2007 LiDAR**
Flying height above ground level meters (AGL)	1100	1000
Absolute Vertical Accuracy in meters	0.063	0.034
Relative Accuracy in meters (calibration)	0.058	0.054
Horizontal Accuracy (1/3300th * AGL) meters	0.37	0.33
Discharge @ time of flight (cms)	13.05	20.8 – 21.8

LiDAR data collection over the Reach 1 field survey location was obtained in a single flight on October 8, 2007 between 1:30 and 6:00 pm. During the LiDAR flight, ground quality control data were collected along roads and other permanent flat surfaces within the collection area. These data were used to adjust for absolute vertical accuracy.

Field Data Acquisition

A river survey crew was dispatched at the soonest possible date (October 13, 2007) after the 2007 flight to collect ground truth data within the Reach 1. The initial aim was to survey water surface elevations at cross sections of the channel, but the survey was limited to near shore measurements due to high velocity conditions. We collected 187 measurements of bed elevation and depth one to fifteen meters from banks along both sides of the channel (Figure 8a) using standard total station longitudinal profile survey methods (Harrelson, 1994). Seventy-six and 98 measurements were collected along the east and west banks, respectively, at intervals of approximately 1 to 2 meters. Thirteen additional measurements were collected along the east bank at approximately ten meter intervals. Depth measurements were added to bed elevations to derive water surface elevations. Discharge during the survey ranged between 22.5 and 22.7 cms during the survey of the east bank and remained steady at 22.5 cms during the survey of the west bank (USGS station 14142500).

LiDAR Processing

The goal of LiDAR processing for this project was to classify LiDAR point data within the active channel as water and output this subset data for further analysis. The LiDAR imagery was first clipped to the active channel using a boundary digitized from the 2006 high resolution orthophotography. LiDAR point data were then reclassified to remove bars, banks, and overhanging vegetation (Figure 11).

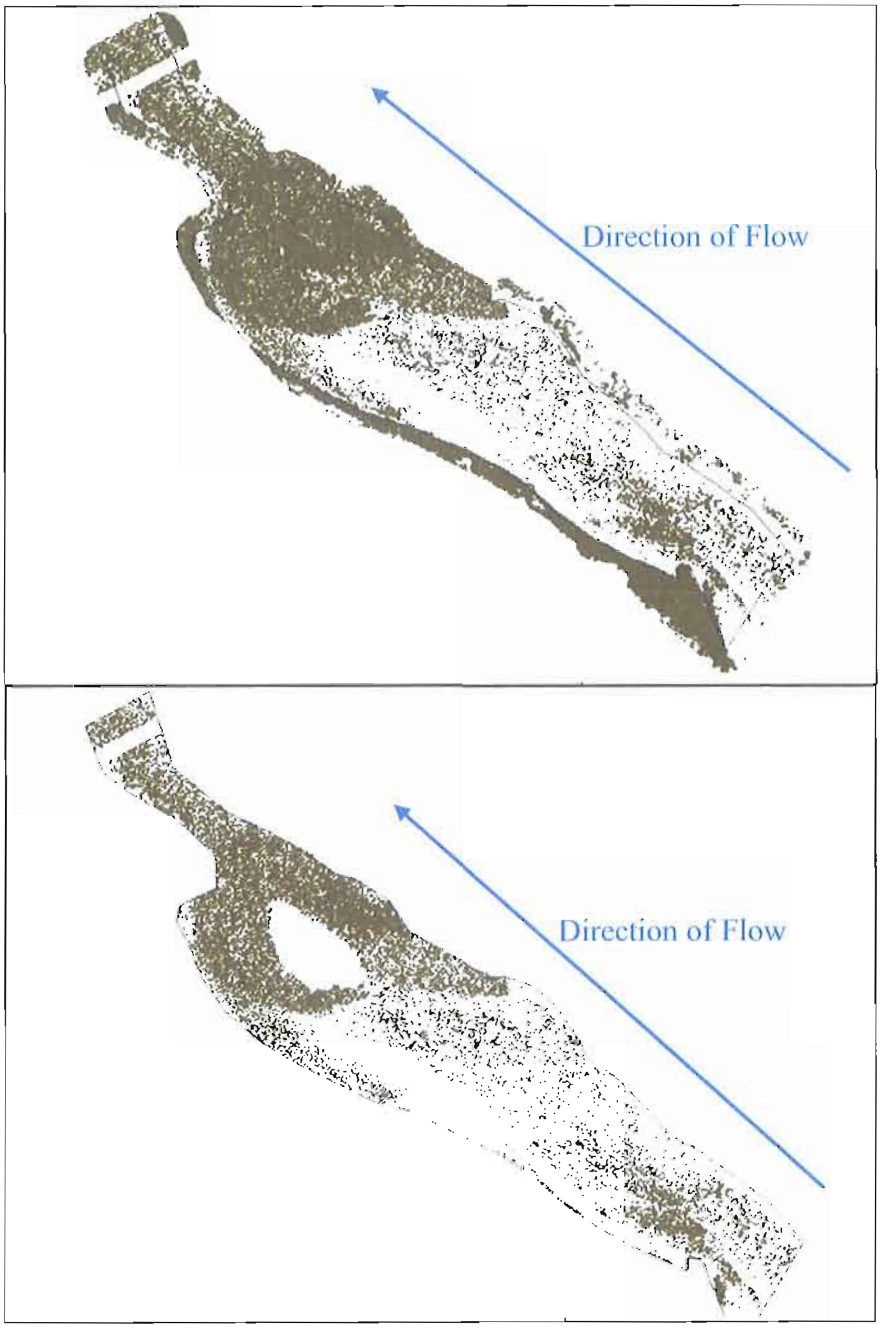


Figure 11. LiDAR Point Filtering Processing Step. LiDAR processing steps. Top image shows entire LiDAR point cloud clipped to active channel boundary. Lower image shows the final processed LiDAR points representing only those points that reflect off the water surface. All bars and overhanging vegetation have been removed as well.

Water points were classified using the ground classification algorithm in Terrascan© (Soininen, 2005) to separate water surface returns from those off of vegetation or other surfaces elevated above the ground. The classification routine uses a proprietary mathematical model to accomplish this task.

Once the ground classification was finished, classified points were visually inspected to add or remove false positives and remove in-channel features such as bar islands. A total of 11,593 of 1,854,219 LiDAR points were classified as water. Points classified as water were output as comma delimited x,y,z ASCII text files (XYZ), then converted to a 0.5 meter linearly interpolated ESRI formatted grid using ESRI geoprocessing model script.

Calculation of Water Surface Slopes

Water surface slopes were calculated using the rise over run dimensionless slope equation where the rise is the vertical difference between upstream and downstream water surface elevations and run is the longitudinal distance between elevation locations.

LiDAR data is typically used in grid format. For this reason grid data were used for calculation of water surface slopes. We used linear interpolation to grid the LiDAR point data as this is the standard method used by the LiDAR contractor. In order to compare the LiDAR and field data it was also necessary to interpolate field measurements to create a water surface for the entire stream. The field data-based DEM was created using kriging interpolation within ArcGIS Desktop Spatial Analyst (Figure 12). No quantitative analysis was performed to evaluate the interpolation method of the field-based water surface. The kriging interpolation was chosen because it producex the smoothest water surface based on visual inspection when compared to linear and natural neighbor interpolations, which generated irregular fluctuations that were unrealistic for a water surface. The kriged surface provided a water surface elevation model for comparative analysis with LiDAR.

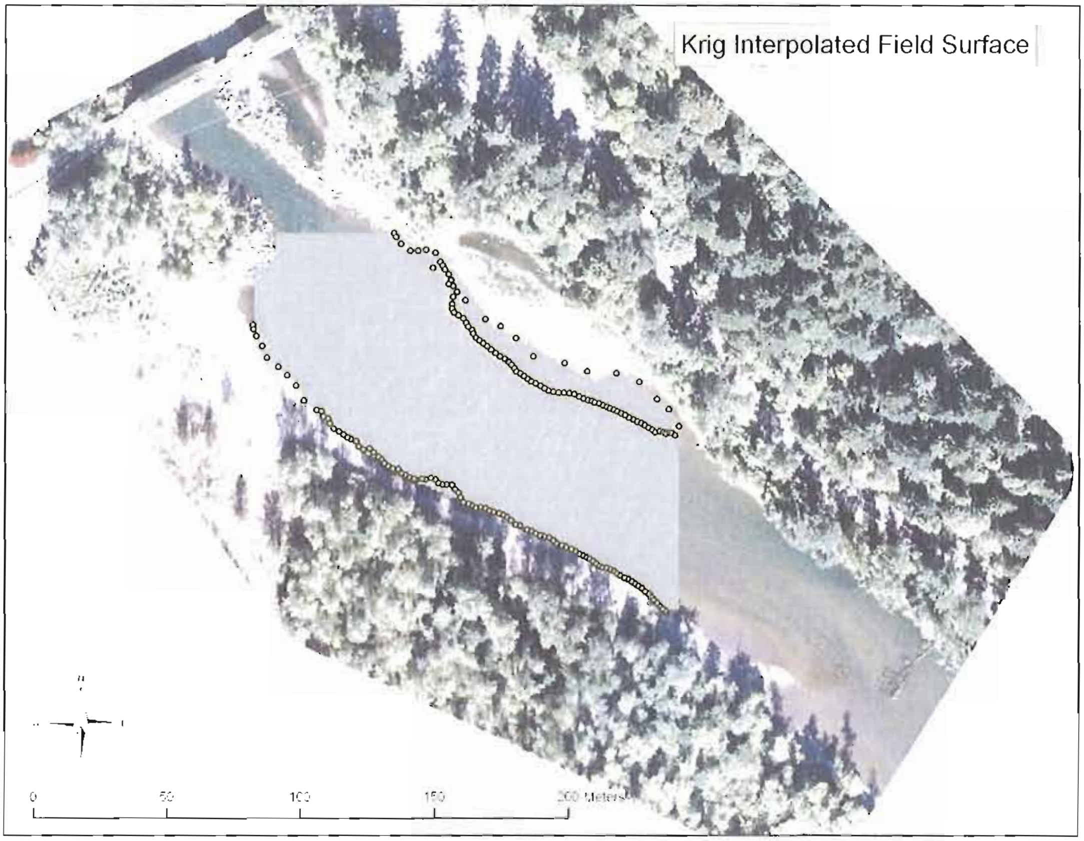


Figure 12. Field DEM Interpolated using Kriging. Field DEM interpolated from field survey points using kriging method found in ArcGIS Spatial Analyst. DEM has been hillshaded to show surface characteristics. The very small differences in water surface elevations generate only slight variations in the hillshadeing.

To compare LiDAR and field-based water surface slopes, water surface elevations from the LiDAR and field-based DEMS were extracted at the same locations along Reach 1. To accomplish this, 37 cross sections were manually constructed at approximately 5m spacings (Figure 13). Cross sections comparisons were used rather than point-to-point comparisons between streamside field and LiDAR data points because the cross sections provide water surface slopes that are more representative of the entire channel. The 5m interval spacing was considered to be a sufficient for fine resolution slope extraction. Because cross section center points were used to calculate the longitudinal distance and because the stream was sinuous, the projection of the cross sections from the center line to the banks led to stream side distances between cross sections that differed from 5m.

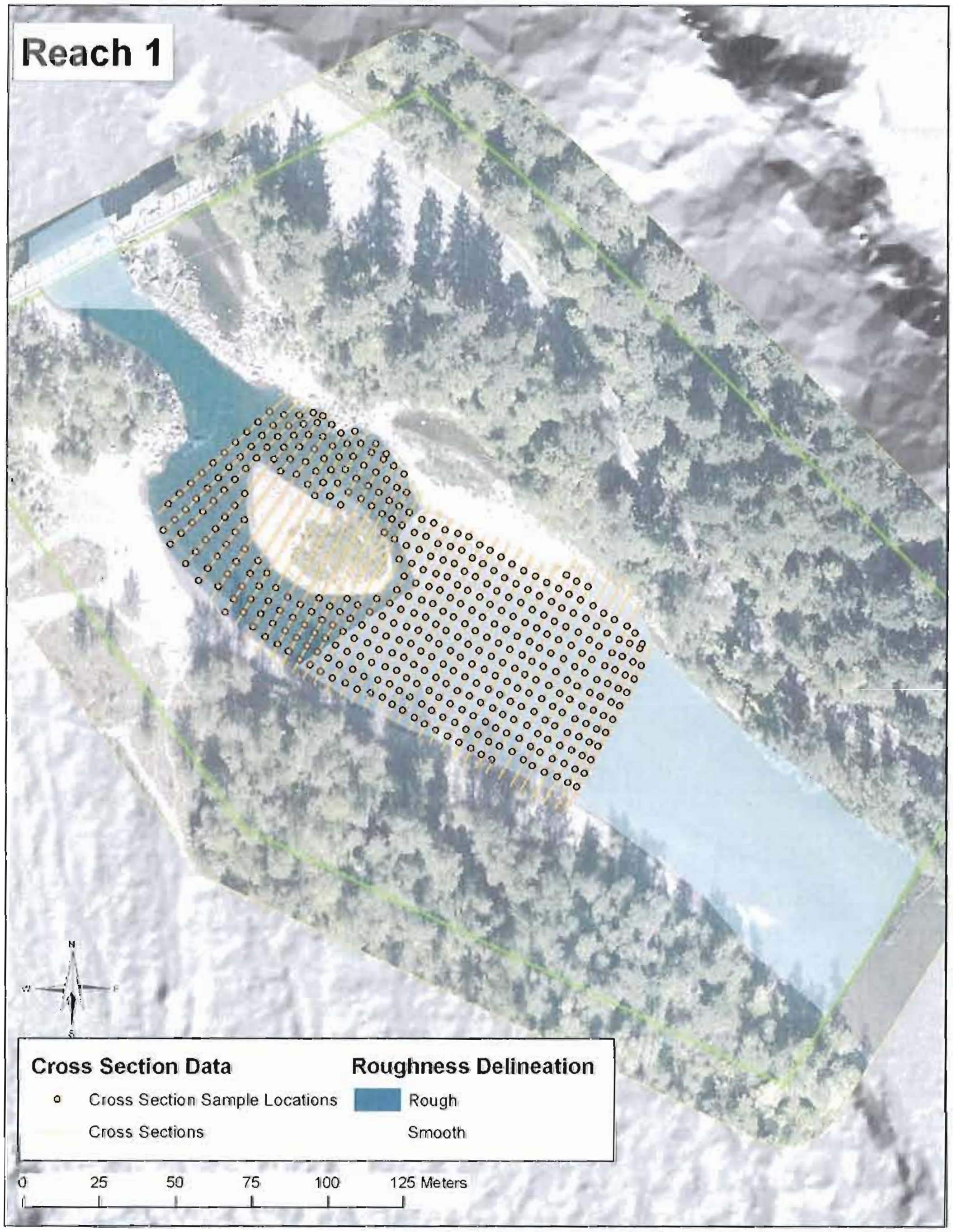


Figure 13. Reach 1 LiDAR Cross Sections and Sample Point Locations. Reach 1 LiDAR-derived cross section sample locations and areas of smooth and rough water surface delineations. 37 cross section and 444 sample points lie within Reach 1.

Cross sections were extracted using a custom ArcObjects VBA script (Appendix A). This script extracted 1 cell nearest neighbor elevations along the transverse cross sections at 5 meter intervals creating 444 cross section sample locations (Figure 13). Cross section averages were calculated using field-based and LiDAR-based elevation water surface grids. The average cross sectional elevation value for field and LiDAR data were then exported to Excel files, merged with longitudinal distance between cross section, and used to calculate field survey-based and LiDAR-based slopes between cross sections.

Reaches 2 and 3, for which only LiDAR data were available, were sampled using the same cross sectional approach used in Reach 1. The data extracted from these reaches were used to characterize how LiDAR-based elevations, slopes and point densities interact with varying water surface roughness. Within Reach 2, 359 cross sections were drawn and elevations were sampled every five meters along each cross section creating 3,456 cross section sample locations (Figure 9). Reach 3 contained 550 cross sections and 3,348 cross section sample locations (Figure 10). Slopes were calculated between each cross section.

Evaluating LiDAR Slope Accuracies and Controls

The accuracy of elevation data is the major control on slope accuracy, so a comparative analysis was performed using field survey and LiDAR elevations. First, field-based and LiDAR slopes were calculated at distance intervals of five, ten and twenty meters using average cross section elevations to test the sensitivity of the slopes to vertical inaccuracies in the LiDAR data. The field and LiDAR elevations were differenced using the same points used to create average cross section elevations. Differences were plotted in the form of histogram and cumulative frequency plot after transforming them into absolute values. Descriptive statistics were calculated to examine the range, minimum, maximum, and mean offset between data sets. Finally LiDAR and field-based values were compared using regression analysis.

This study also examined the effects of water surface roughness on LiDAR elevation measurements, LiDAR point density, and LiDAR derived water surface slopes. Each reach was divided into smooth and rough sections based on visual analysis of the orthophoto data. One-meter resolution slope rasters were created from the LiDAR water surface grids using ArcGIS Spatial Analyst. One meter resolution point density grids were created from LiDAR point data (ArcGIS Spatial Analyst). Using the cross section sample points, values for water surface type, elevation, slope, and point density were extracted within each reach. Point sample data were transferred to tabular format, and average values were generated for each cross section. These tables were used to calculate descriptive statistics associated with water surfaces such as elevation variance, average slope variance, average point density, and average slope.

It is assumed in this study that smooth water surfaces are associated with pools and thus ought to have relatively low slopes. Conversely rough water surfaces are assumed to be representative of riffles and rapids, and thus ought to have relatively steeper slopes. Reach 1 contains field data, so slopes from LiDAR and field data were compared with respect to water surface conditions as determined from the aerial photos.

CHAPTER V

RESULTS

Results of this study encompass three analyses. Elevation analysis describes the statistical difference between LiDAR and field-based water surface elevations for Reach 1. Slope analysis compares LiDAR derived and field-based slopes calculated at 5, 10, and 20m longitudinal distances. These analyses aim to quantify both slope accuracy and slope sensitivity. Lastly, water surface analysis examines the relationship between LiDAR measured water surface slopes, point density, and water surface roughness.

Comparison of Absolute Elevations from Field and LiDAR Data in Reach 1

The difference between water surface elevations from LiDAR affects the numerator within the rise over run equation, which in turn affects slope. This elevation analysis evaluation quantifies differences between field and LiDAR data. LiDAR-based cross section elevations were differenced from field-based cross section elevations. Difference values were examined through statistical analysis.

In terms of absolute elevations relative to sea level, the majority of LiDAR-based water surface elevations were lower than field-based elevations, although the LiDAR elevations were higher in the upper portion of Reach 1. Differences ranged between -0.04 and 0.05m with a mean absolute difference between field and LiDAR elevations of 0.02m (Figure 14 and Table 2). The range of differences is within the expected relative accuracies of LiDAR claimed by the LiDAR provider. Elevations for field and LiDAR data are significantly correlated with an R^2 of 0.94 (Figure 15).

The negative offset was expected given that discharge at time of LiDAR acquisition was lower than discharge at time of field data acquisition. Discharge during field acquisition ranged between 22.5 and 22.7 cfs, while discharge during LiDAR acquisition was between 20.8 and 21.8cfs. The portion of Reach 1 where LiDAR water surface measurements were higher than field measurements may be related to difference in discharge or change in bed configuration. Overall results showed that LiDAR data and field-based water surface measurements are comparable.

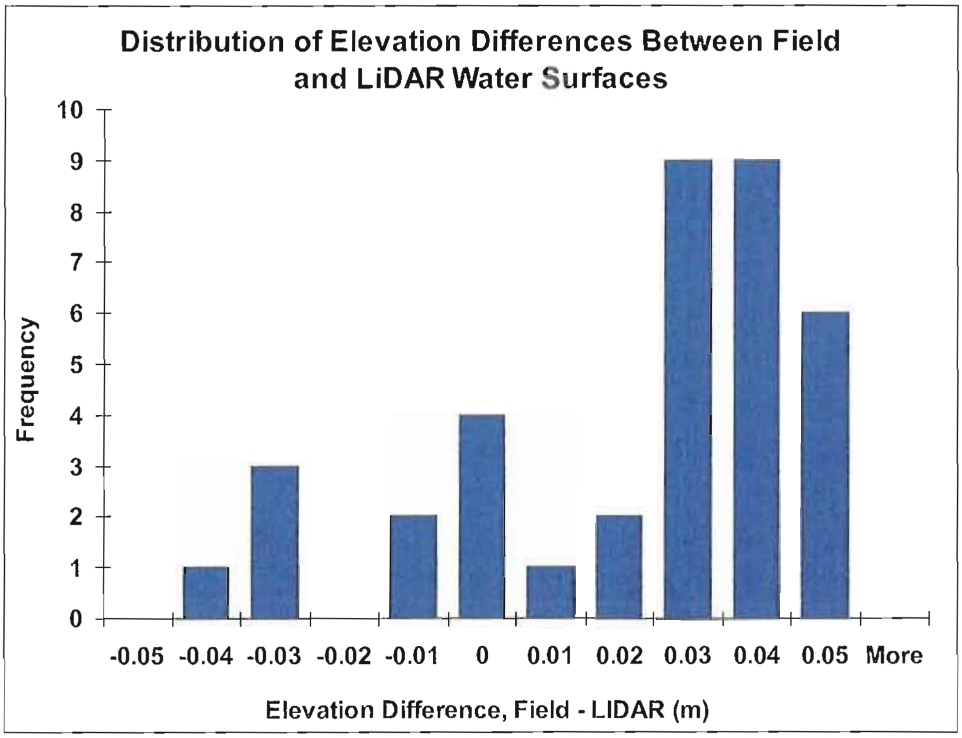


Figure 14. Differences Between LiDAR and Field Based Elevations. Elevation difference statistics between cross sections derived from field and LiDAR elevation data. Positive differences indicate that field-based elevations were higher than LiDAR; negative differences indicate LiDAR elevations were higher. Values on x axis represent minimum difference within range. For example, the 0.01 category includes values ranging from 0.01 to 0.0199.

Table 2. Results of LiDAR and Field Elevation Comparison. Descriptive and regression statistics for absolute difference |Field – LiDAR| values between cross section elevations. All units in meters. Sample size is 37.	
Mean	0.028
Median	0.030
Standard Deviation	0.013
Kurtosis	-0.640
Skewness	-0.484
Range of difference	0.093
Minimum difference	0.002
Absolute maximum difference	0.047
Confidence Level(95.0%) (m)	0.004

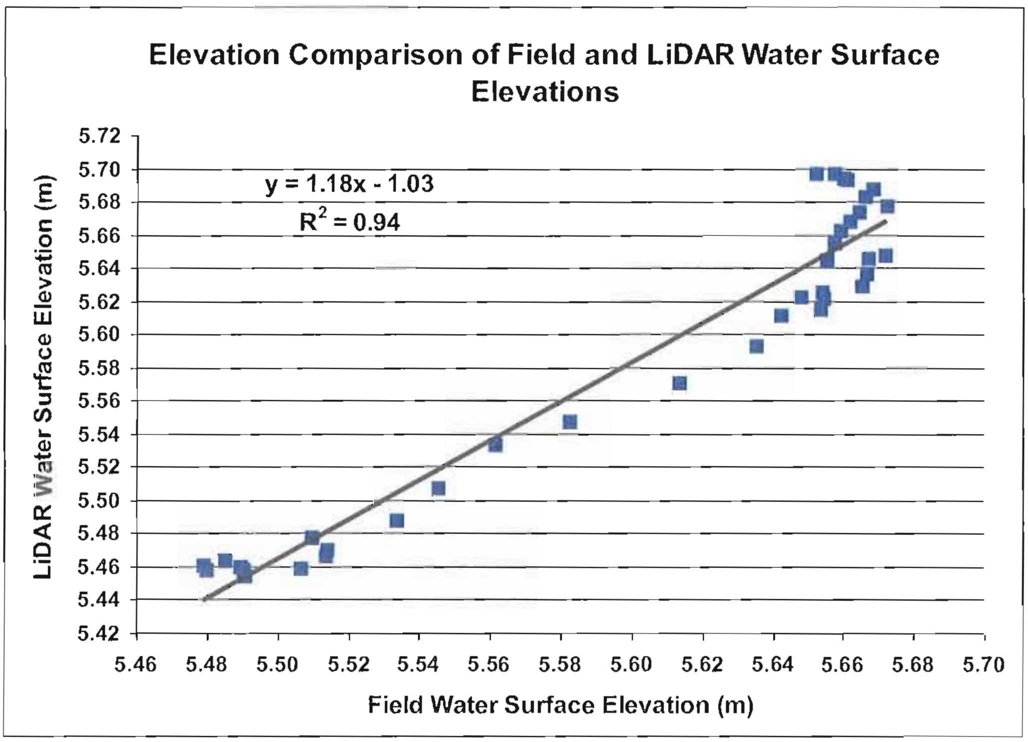
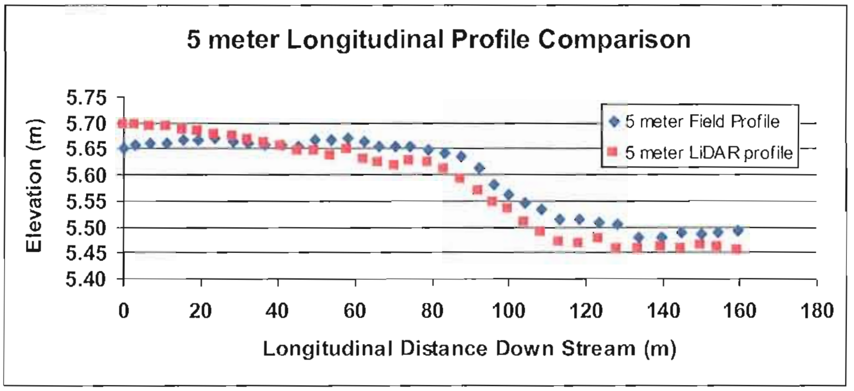
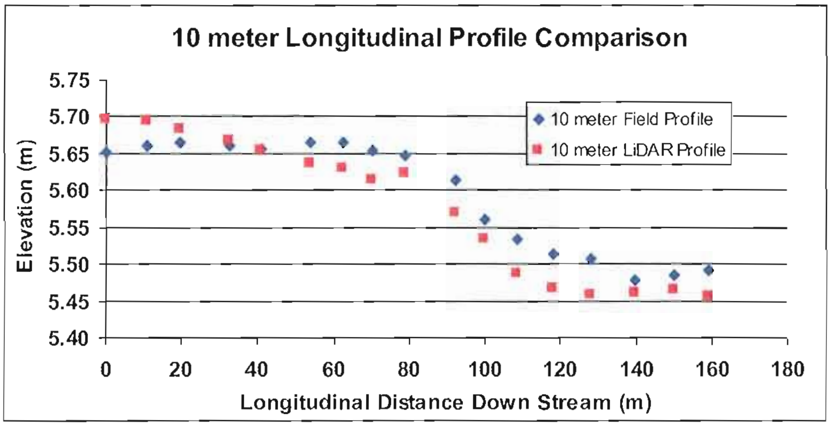


Figure 15. Regression of LiDAR and Field Cross Section Elevations.
Regression of field-based (x) and LiDAR-based (y) cross section elevations.

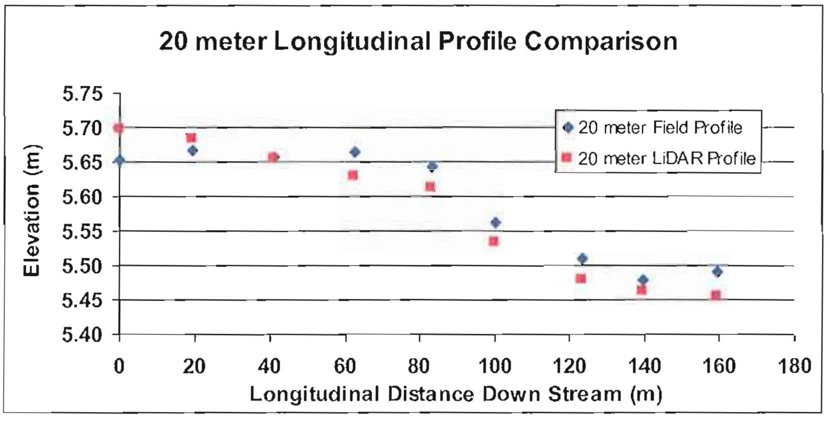
Comparison of longitudinal profiles of field and LiDAR water surfaces shows a clear relationship in overall shape (Figure 16), capturing similar trends in longitudinal profiles. Figure 16 shows field and LiDAR profiles become more similar in shape as distance between cross sections increases. In terms of overall shape, the greatest differences occur in the upper 30 m, where LiDAR-based profiles demonstrate a higher slope than do field-based measurements. Because of the five day lag between LiDAR and field measurements in this mobile bed stream, it is impossible to know the degree to which this difference represents error in measurements or real change in the system.



A



B



C

Figure 16. Comparison of LiDAR and Field Longitudinal Profiles (5, 10, 20 meters). Longitudinal profiles of a) 5 meter, b) 10 meter, and c) 20 meter cross section elevations.

Slope Comparisons

Slope in this study is calculated as the dimensionless ratio of rise over run. As noted in the Methods section, slopes were calculated over three different horizontal intervals to test the sensitivity of the LiDAR's internal relative accuracy.

Differences in 5m LiDAR and field-based slopes derived from cross sections reveal substantial scatter (Figure 17a), although they clearly covary. Ten meter interval slopes show a stronger relationship (Figure 17b), while slopes based on cross sections spaced 20 m apart have the strongest relationship (Figure 17c). The slope associated with regression of field and LiDAR elevation data is not approximately 1 as one might expect. This is because LiDAR elevations are higher than field elevations at the upstream end of the reach, and lower at the downstream end.

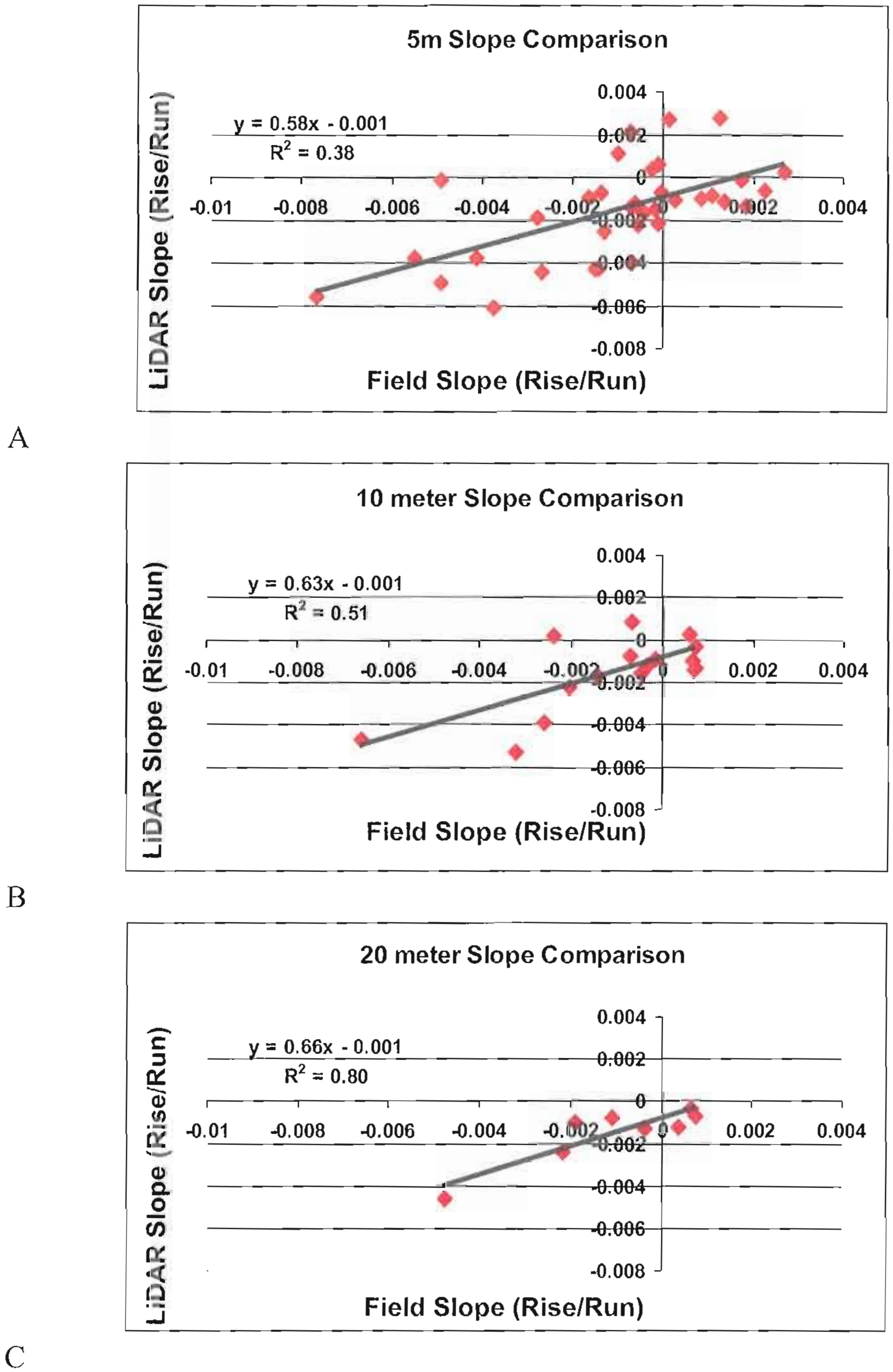
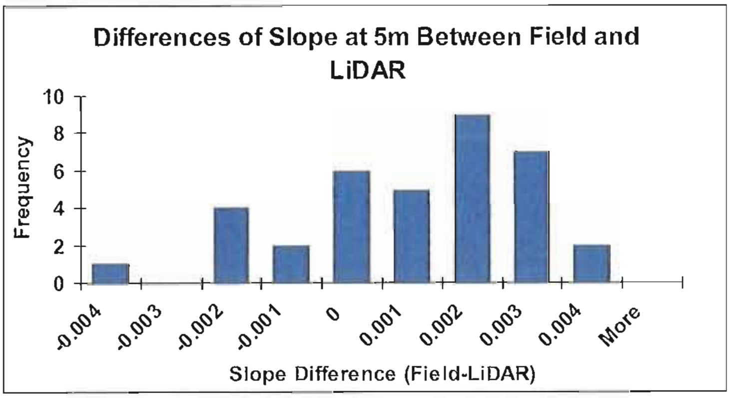
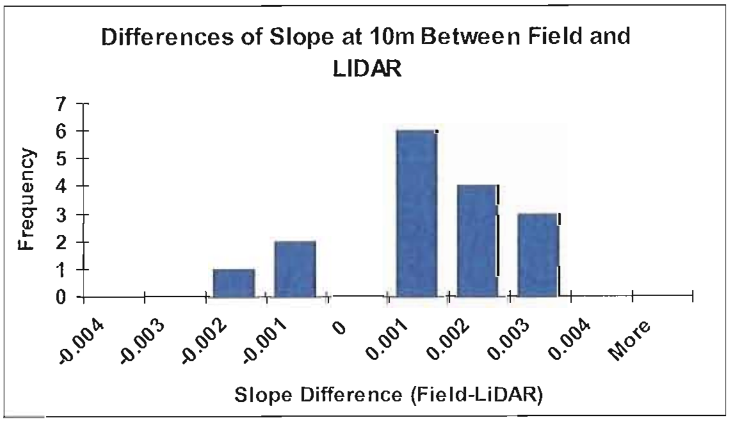


Figure 17. Regression of Field and LiDAR Based Slopes (5, 10, 20 meters). Scatter plots showing comparisons between slope values calculated at distance intervals of a) 5 meters, b) 10 meters, and c) 20 meters.

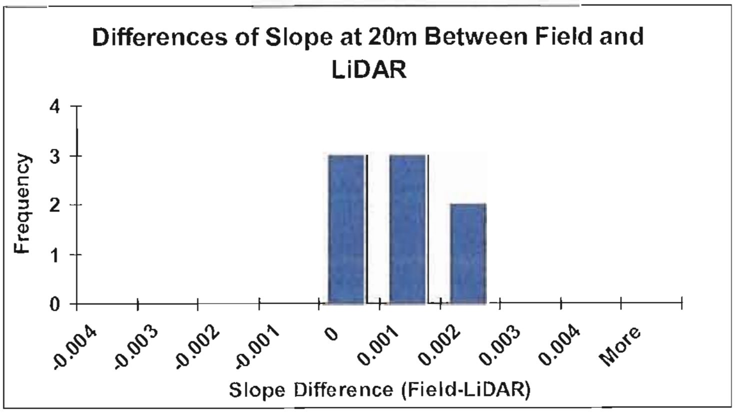
Figure 18 shows how the range of differences between LiDAR and field-based water surface slopes decrease as longitudinal distance increases. Five meter slope differences ranged between -0.004 and 0.004 (Figure 18a). Ten meter slope differences ranged between -0.002 and 0.003 (Figure 18b). Twenty meter slope differences ranged between 0 and 0.002 (Figure 18c).



A



B



C

Figure 18. Differences Between LiDAR and Field Based Slopes (5, 10, 20 meters). Histogram charts showing difference values between field and LiDAR derived slopes at a) 5 meter slope distances, b) 10 meter slope distances, and c) 20 meter slope distances.

The mean difference between slopes decreases from 0.0017 to 0.0007 as slope distance interval is increased. Maximum slope difference and standard deviation of offsets decrease from 0.001 to 0.0005 and 0.0047 to 0.0014 respectively. Regression analysis of these data show a significant relationship for all three comparisons, and adjusted R^2 increased from 0.357 to 0.763 with slope distance interval (Table 3).

Table 3. Results of LiDAR and Field Slope Comparison (5, 10, 20 meters). Descriptive and regression statistics for offsets between field and LiDAR derived slope values (Field minus LiDAR). Slope values are dimensionless rise / run. All data is significant at 0.01.

Distance Interval	**5m**	**10m**	**20m**
Mean	0.0017	0.0012	0.0007
Standard Deviation	0.0010	0.0007	0.0005
Range of Difference	0.0080	0.0047	0.0024
Minimum difference	0.0000	0.0000	0.0001
Maximum difference	0.0047	0.0026	0.0015
Count	36	16	8
Adjusted R squared	0.36	0.47	0.76

Water surface slope for the entire length of Reach 1 (159.32m) was compared and yielded a difference of 0.0005. This difference is smaller (by 0.0002) than the difference between 20 meter slope (Table 4). Slope was calculated by differencing the most upstream and downstream cross sections and dividing by total length of reach. Differences between LiDAR and field-based slopes may represent real change due to the five day lag between data sets and difference in discharge.

Table 4. Results of Reach 1 Slope Comparison. Comparison of slopes calculated using the farthest upstream and downstream cross section elevation values. Slope values have dimensionless units stemming from rise over run.

	Upper Elevation (m)	**Lower Elevation (m)**	**Reach Length (m)**	**Slope**
Field	5.652	5.491	159.32	-0.0010
LiDAR	5.697	5.455	159.32	-0.0015

Surface Roughness Analysis

Water surface condition was characterized as smooth or rough based on 2006 aerial photography (Figure 19). Surface roughness was examined to understand its effect on LiDAR data within the active channel, as well as LiDAR's ability to potentially capture difference in water surface turbulence. Table 5 shows statistics with relation to water surface condition for all three reaches.

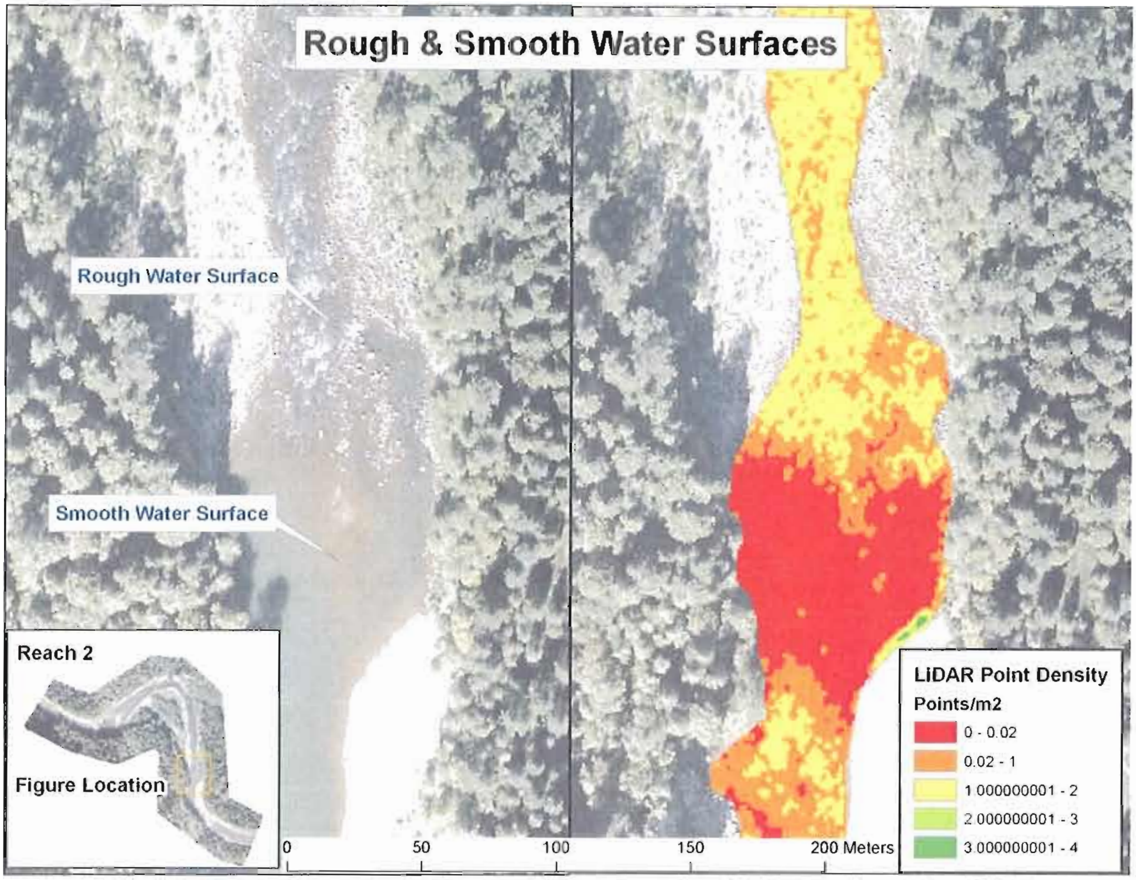


Figure 19. Relationship of Water Surfaces to LiDAR Point Density. 2006 aerial photos were used to delineate rough and smooth water surfaces. Image on left shows a transition between rough water surface (seen as white water) and smooth water surface (seen as upstream pool). Image on right shows LiDAR point density in points per square meter.

In all reaches point density, variance of elevations, and water surface slopes were significantly higher in rough surface conditions. These results indicate that LiDAR point density is directly related to the roughness of a water surface and that is capturing the rough water characteristics one would expect in areas where turbulence generates surface waves.

Table 5. Water Surface Roughness Results for Reach 1, 2, and 3. Water surface statistical output for rough and smooth water surface of Reaches 1, 2, and 3. Results within table represent average values for each Reach. Slope values have dimensionless units from rise over run equation derived from ESRI generated slope grid. Point density values based on points/m^2. Elevation variance in meters.

	Reach 1	**Reach 2**	**Reach 3**
Rough water			
No. of Sample Points	153	1981	1968
Avg Slope	-0.013	-0.011	-0.007
Point Density (pts/m^2)	1.195	1.002	1.217
Elevation Variance (m)	0.003	0.018	0.041
Smooth water			
No. of Sample Points	290	1474	1378
Avg Slope	0.0075	-0.0006	-0.0033
Point Density (pts/m^2)	0.149	0.550	0.480
Elevation Variance (m)	0.001	0.0077	0.024

Within Reach 1, cross section elevations were separated into rough and smooth water conditions and slopes were calculated using field and LiDAR data sets (Table 6). Again, results showed that rough water surfaces have greater slopes than smooth water surfaces. The smooth water surface of Reach 1 yielded a larger discrepancy between field and LiDAR derived slopes compared to rough water surface. This is because small differences between LiDAR and field elevations generate larger proportional error in the rise / run equation when total elevation differences between upstream and downstream are small.

Table 6. Results of Reach 1 Water Surface Roughness Comparison. Reach 1 water surface roughness slope analysis. Reach 1 was divided into smooth and rough water surfaces based upon visual characteristics present in aerial photography. Slopes were calculated for each area and compared with field data to examine accuracy.

	Surface Type	**Reach Length (m)**	**Upper Elevation (m)**	**Lower Elevation (m)**	**Slope**	**Slope Difference**
Field	Smooth	83.11	5.652	5.642	-0.0001	N/A
LiDAR	Smooth	83.11	5.697	5.612	-0.0010	0.0009
Field	Rough	71.73	5.635	5.491	-0.0020	N/A
LiDAR	Rough	71.73	5.592	5.455	-0.0019	-0.0001

Prior to collections of the 2007 data, Reach 3 contained the former Marmot Dam that was dismantled on October 19th, 2007 (Figure 20). The areas at and directly below the dam are rough water surfaces. The super critical flow at the dam yielded a slope of -0.896 (Table 7). The run below the dam contained low slope values of less than -0.002. Both the dam fall and adjacent run yielded high point densities of greater than 2 points per square meter.

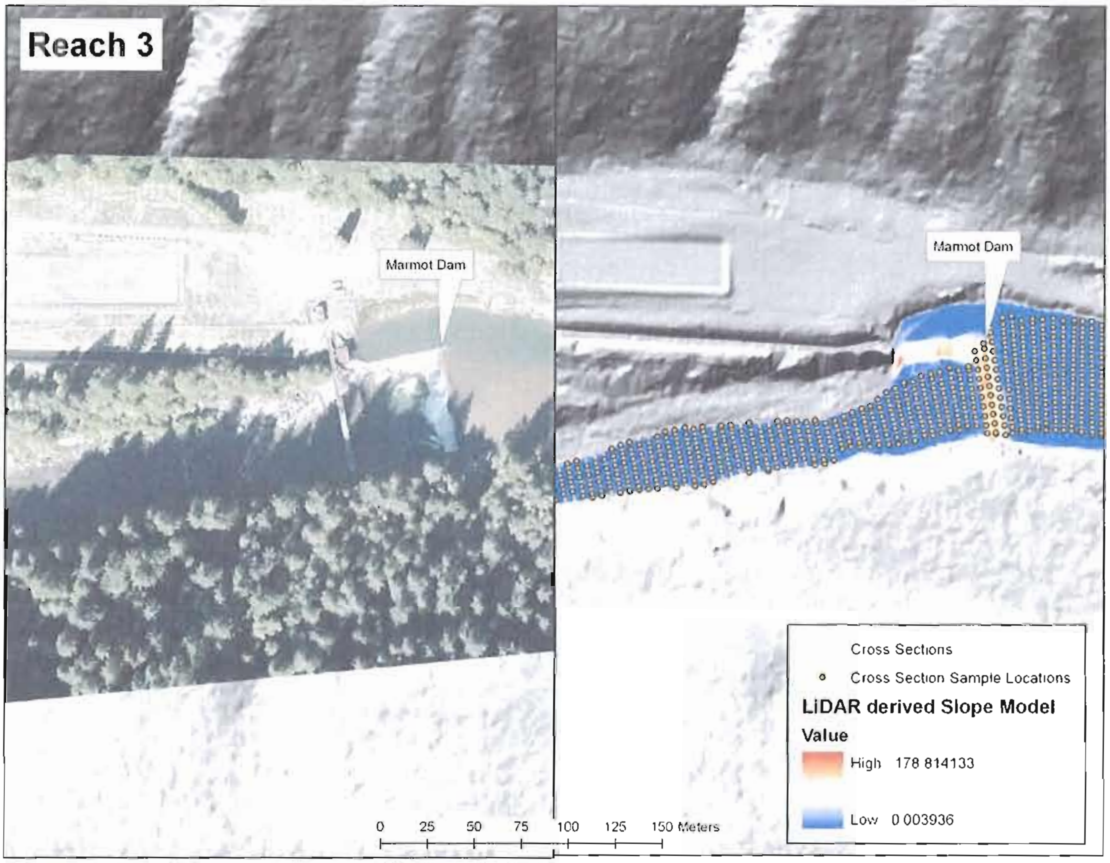


Figure 20. Marmot Dam: Orthophotography and Colorized Slope Model. Marmot Dam at far upstream portion of Reach 3. Image on left shows dam site in 2006 orthophotography. Image on right shows the increase in slope associated with the dam. Marmot Dam was removed Oct. 19th, 2007.

Table 7. Subset of Reach 3 Water Surface Roughness Analysis Near Marmot Dam. Subset of Reach 3 immediately surrounding Marmot Dam roughness analysis containing values for Marmot Dam. The roughness results fell within expectations showing increases in slope at the dam fall and high point densities at the dam fall and immediate down stream run.

Habitat Type	Avg Slope	Point Density	Point Density Variance
Dam Fall	-0.896	2.284	1.003
Dam Run	-0.001	2.085	5.320

CHAPTER VI

DISCUSSION

The elevation analysis portion of this study shows that LiDAR can provide water surface profiles and slopes that are comparable to field-based data. The differences between LiDAR and field based measurements can be attributed to three potential sources. The first is the relative accuracy of the LiDAR data which has been reported between 0.05m and 0.06m by the vendor. The second source can be associated with the accuracy of field based measurements which are similar to the relative accuracy of the LiDAR (0.03m-0.05m). Lastly, the discharge differed between field data collection and LiDAR collection by 0.02cms. It is possible that much of the 0.05m difference observed through most of the Reach 1 profile (Figure 16) could be attributed to the difference in discharge and changes in bed configuration, but without further evidence, the degree of difference due to error or real change cannot be identified. Even if one attributes all the difference to error in LiDAR measurements, the overall correspondence of LiDAR and field measurement (Figure 15 and 16) indicates that LiDAR-based surveys are useful for many hydrologic applications.

In the upper portion of the reach, the profiles display LiDAR elevations that are higher than the field data elevations, whereas the reverse is true at the base of the reach. This could be a function of difference in discharge between datasets, change in bed configuration, or an artifact of low point density. Low density of points forces greater lengths of interpolation between LiDAR points leading to a coarse DEM (Figure 21). Overall, the analysis Reach 1 profile indicates that LiDAR was able to match the field-based elevation measurements within ±0.05m.

Figure 21. LiDAR Point Density versus Interpolation. Side by side image showing long lines of interpolation associated with smooth water surfaces (right image). Smooth water surfaces tend to have low LiDAR point density. The image on the right shows a hillshade of the LiDAR DEM. The DEM has been visualized using a 2 standard deviation stretch to highlight long lines of interpolation.

The comparability of LiDAR and field-based slopes showed a significant trend with increasing downstream distances between cross sections. Adjusted R^2 values increased from 0.36 to 0.76 and the range of difference between field and LiDAR based slopes decreased from 0.0047 to 0.0014 as longitudinal distance increased from 5 to 20-

m. This suggests that the 0.05m of expected variation of LiDAR derived water surface elevation has less effect on water surface slope accuracy as distance between elevation measurements points increases. Likewise, slopes accuracies along rivers with low gradients will improve as the longitudinal distance between elevation points increases.

Overall, data has shown that LiDAR can measure water surface slopes with mean difference relative to field measurements of 0.017, 0.012, and 0.007 at horizontal distances of 5, 10, and 20 meters respectively. Although the discrepancy between field and LiDAR-based slopes is greatest at 5-m intervals, the overall slopes (Fig 17) and longitudinal profiles (Fig 16) even at this distance generally correspond. The use of a 5m interval water surface slope as a basis for comparison is really a worst case example, as water surface slopes are usually measured over longer reach scale distances where the discrepancy between LiDAR and field-based measurements is lower. The continuous channel coverage and accuracies derived from LiDAR represent a new level of accuracy and precision in terms of spatial extent and resolution of water surface slope measurements.

Analysis of surface roughness found that rough water surfaces had significantly higher point densities than smooth water surfaces. Rough water surfaces averaged at least 1 point/m^2, while smooth water surfaces averaged less than 1 point/2m^2. Longitudinal profiles of Reach 1 indicate the most accurate water surface measurements occur in areas of higher point density (Fig. 16). Future applications that attempt to use

LiDAR to measure water surface slope ought to sample DEM elevations from high point density areas of channel.

Water surface analysis also showed trends relating water surface roughness and slope. Rough water surfaces for all three analysis reaches averaged larger average slope values than smooth water surfaces. This is because rough water surfaces are commonly associated with steps, riffles, and rapids. All three of these habitat types are areas have higher slopes than smooth water habitats. Smooth water surfaces are commonly associated with pools or glides, which would be areas of lower slope. Future research should examine the potential for using LiDAR to characterize stream habitats based on in-stream point density and slope.

This study is not without its limitations. The field area used to test the accuracy of LiDAR is only representative of a small portion of the Sandy River. Comparisons of field and LiDAR data would be improved by having mid-channel field data. One might also question the use of field based water surface slopes as control for measuring "accuracy". Water surface slope is difficult to measure for reasons stated earlier in this paper. One might make the argument that there is no real way to truly measure LiDAR accuracy of water surface slope, and that LiDAR and field based measurements are simply comparable. In this context, LiDAR holds an advantage over field based measurements given its ability to measure large sections of river in a single day.

LiDAR has a distinct advantage over traditional methods of measurement in that measurements are returned from the water surface, and consequently not subject to errors associated with variability of surface turbulence piling up against the measuring device. LiDAR can also capture long stretches of channel within a few seconds reducing the influence of changes in discharge. LiDAR data in general does have its limitations. LiDAR data are only as accurate as the instrumentation and vendor capabilities. LiDAR must be corrected for calibrations and GPS drift to create a reliable data set, and not all LiDAR vendors produce the same level of quality.

LiDAR data may be more accurate in some river reaches than others. The study reaches of this study contained well defined open channels, which made identifying LiDAR returns off the water surface possible. Both LiDAR data sets were collected at low flows. Flows that are too low or channels that are too narrow may limit ability to extract water surface elevations because of protruding boulders or dense vegetation that hinders accurate measurements. In some cases vegetation within and adjacent to the channel may interfere with LiDAR's ability to reach the water surface. Researchers should consider flow, channel morphology, and biota when obtaining water surface slopes from LiDAR.

CHAPTER VII

CONCLUSION

This paper examined the ability of LiDAR data to accurately measure water surface slopes. This study has shown that LiDAR data provides sufficiently accurate elevation measurements within the active channel to accurately measure water surface slopes. Measurement of water surface slope with LiDAR provides researchers a tool which is both more efficient and cost effective in comparison with traditional field-based survey methods. Additionally, analysis showed that LiDAR point density is significantly higher in rough surface conditions. Water surface elevations should be gathered from high point density areas as low point density may hinder elevation accuracy. Channel morphology, gradient, flow, and biota should be considered when extracting water surface slopes as these attributes influence water surface measurement. Further study should examine accuracy of LiDAR derived water surface slopes in channel morphologies other than those in this study. Overall, the recognition that LiDAR can accurately measure water surface slopes allows researchers an unprecedented ability to study hydraulic processes for large stretches of river.

APPENDIX ARCGIS VBA SCRIPT CODE

Common:

```
Public g_pStrmLayer As ILayer          ' stream centerline layer selected by user (for step 1)
Public g_StreamLength As Double        ' stream centerline length (for step 1)
Public g_InputDistance As Integer 'As Double      ' distance entered by user (for step 1)
Public g_NumSegments As Integer        ' number of sample points entered by user (for step 1)
Public g_pPointLayer As ILayer         ' point layer created from stream centerline (for step 1)
Public g_PntShpFlName As String        ' point layer pathname (for step 1)
Public g_pMouseCursor As IMouseCursor  ' mouse cursor
Public g_LinearConverson As Double     ' linear conversion factor
Public g_pDEMLayer As IRasterLayer     ' DEM layer (for steps 3 and 4)
Public g_DEMConvertUnits As Double     ' DEM vertical units conversion factor (for steps 3 and 4)
Public g_MaxSearchDistance As Double   ' maximum search distance (for step 4)
Public g_NumDirections As Integer      ' number of directions to search in (for step 4)
Public g_SampleDistance As Double      ' sample distance (for step 5)
Public g_SampleNumber As Double        ' total sample points (for step 5)
Public g_VegBeginPoint As Boolean      ' where to start the calucaltion (for step 5)
Public g_VegCaclMethod As Boolean      ' which method for Vegetation Calculation (for step 5)
Public g_pContribLayer As ILayer       ' contributing point layer (for step 6)
Public g_pReceivLayer As ILayer        ' receiving point layer (for step 6)
Public g_pOutputLayerName As String    ' output shapefile (for step 6)

Function VerifyField(fLayer As ILayer, fldName As String) As Boolean
  ' verify that topo fields are in the stream centerline point layer
  Dim pFields As IFields
  Dim pField As IField
  Dim pFeatLayer As IFeatureLayer
  Dim pFeatClass As IFeatureClass
  Set pFeatLayer = fLayer
  Set pFeatClass = pFeatLayer.FeatureClass
  Set pFields = pFeatClass.Fields
  For i = 0 To pFields.FieldCount - 1
    Set pField = pFields.Field(i)
    'MsgBox pField.Name
    If pField.Name = fldName Then
      VerifyField = True
      Exit Function
    End If
  Next
  VerifyField = False
End Function
```

```
Function CalcPointLatLong(inPnt As IPoint, inLayer As ILayer) As IPoint

    ' in point layer
    Dim pFLayer As IFeatureLayer
    Set pFLayer = inLayer

    ' spatial reference environment
    Dim pInSpatialRef As ISpatialReference
    Dim pOutSpatialRef As ISpatialReference
    Dim pGeoTrans As IGeoTransformation
    Dim pInGeoDataset As IGeoDataset
    Set pInGeoDataset = pFLayer
    Dim pSpatRefFact As ISpatialReferenceFactory
    ' get map units of shapefile spatial reference
    Dim pPCS As IProjectedCoordinateSystem
    Set pPCS = pInGeoDataset.SpatialReference
    'set spatial reference environment
    Set pSpatRefFact = New SpatialReferenceEnvironment
    Set pInSpatialRef = pInGeoDataset.SpatialReference
    'MsgBox pInSpatialRef.Name
    Set pOutSpatialRef = pSpatRefFact.CreateGeographicCoordinateSystem(esriSRGeoCS_WGS1984)
    Set pGeoTrans =
pSpatRefFact.CreateGeoTransformation(esriSRGeoTransformation_NAD1983_To_WGS1984_1)
    Dim pOutGeom As IGeometry2

    Set CalcPointLatLong = New Point
    Set CalcPointLatLong.SpatialReference = pInSpatialRef
    CalcPointLatLong.PutCoords inPnt.X, inPnt.Y
    Set pOutGeom = CalcPointLatLong
    pOutGeom.ProjectEx pOutSpatialRef, esriTransformForward, pGeoTrans, 0, 0, 0
    'MsgBox inPnt.X & "   " & inPnt.Y & vbCrLf & CalcPointLatLong.X & "   " & CalcPointLatLong.Y

End Function

Sub OpenGxDialog()
    Dim pGxdial As IGxDialog
    Set pGxdial = New GxDialog
    pGxdial.ButtonCaption = "OK"
    pGxdial.Title = "Create Stream Centerline Point Shapefile"
    pGxdial.RememberLocation = True
    Dim pShapeFileObj As IGxObject

    Dim pGxFilter As IGxObjectFilter
    Set pGxFilter = New GxFilterShapefiles 'e.g shp
    Set pGxdial.ObjectFilter = pGxFilter

    If pGxdial.DoModalSave(ThisDocument.Parent.hWnd) Then
        Dim pLocation As IGxFile
        Dim fn As String
```

```
      Set pLocation = pGxdial.FinalLocation
      fn = pGxdial.Name
    End If
    If Not pLocation Is Nothing Then
      g_PntShpFlName = pLocation.Path & "\" & fn
      frm1B.tbxShpFileName.Text = g_PntShpFlName
      frm1B.cmdOK.Enabled = True
    End If
End Sub

Function GetAngle(pPolyline As IPolyline, dAlong As Double) As Double
    Dim pi As Double
    pi = 4 * Atn(1)
    Dim dAngle As Double
    Dim pLine As ILine
    Set pLine = New Line
    pPolyline.QueryTangent esriNoExtension, dAlong, False, 1, pLine
    ' convert from radians to degrees
    dAngle = (180 * pLine.Angle) / pi
    ' adjust angles
    ' ESRI defines 0 degrees as the positive X-axis, increasing counter-clockwise
    ' Ecology references 0 degrees as North, increasing clockwise
    If dAngle <= 90 Then
      GetAngle = 90 - dAngle
    Else
      GetAngle = 360 - (dAngle - 90)
    End If
End Function

Function FeatureExists(strFeatureFileName As String) As Boolean

On Error GoTo ErrHandler:

    Dim pWSF As IWorkspaceFactory
    Set pWSF = New ShapefileWorkspaceFactory

    Dim pFeatWS As IFeatureWorkspace
    Dim pFeatDS As IFeatureClass

    Dim strWorkspace As String
    Dim strFeatDS As String
    strWorkspace = SplitWorkspaceName(strFeatureFileName) & "\"
    strFeatDS = SplitFileName(strFeatureFileName)

    If pWSF.IsWorkspace(strWorkspace) Then
      Set pFeatWS = pWSF.OpenFromFile(strWorkspace, 0)
      Set pFeatDS = pFeatWS.OpenFeatureClass(strFeatDS)
    End If
```

```
    FeatureExists = (Not pFeatDS Is Nothing)

    Set pWSF = Nothing
    Set pFeatWS = Nothing
    Set pFeatDS = Nothing

 Exit Function

 ErrHandler:
    FeatureExists = False

 End Function

'Returns a Workspace given for example C:\temp\dataset returns C:\temp
Function SplitWorkspaceName(sWholeName As String) As String
   On Error GoTo ERH
   Dim pos As Integer
   pos = InStrRev(sWholeName, "\")
   If pos > 0 Then
      SplitWorkspaceName = Mid(sWholeName, 1, pos - 1)
   Else
      Exit Function
   End If
   Exit Function
ERH:
   MsgBox "Workspace Split:" & Err.Description
End Function

'Returns a filename given for example C:\temp\dataset returns dataset
Function SplitFileName(sWholeName As String) As String
   On Error GoTo ERH
   Dim pos As Integer
   Dim sT, sName As String
   pos = InStrRev(sWholeName, "\")
   If pos > 0 Then
      sT = Mid(sWholeName, 1, pos - 1)
      If pos = Len(sWholeName) Then
         Exit Function
      End If
      sName = Mid(sWholeName, pos + 1, Len(sWholeName) - Len(sT))
      pos = InStr(sName, ".")
      If pos > 0 Then
         SplitFileName = Mid(sName, 1, pos - 1)
      Else
         SplitFileName = sName
      End If
   End If
   Exit Function
ERH:
```

```
    MsgBox "Workspace Split:" & Err.Description
End Function

Public Sub BusyMouse(bolBusy As Boolean)
'Subroutine to change mouse cursor

  If g_pMouseCursor Is Nothing Then
    Set g_pMouseCursor = New MouseCursor
  End If

  If bolBusy Then
    g_pMouseCursor.SetCursor 2
  Else
    g_pMouseCursor.SetCursor 0
  End If
End Sub

Function MakeColor(lRGB As Long) As IRgbColor
    Set MakeColor = New RgbColor
    MakeColor.RGB = lRGB
End Function

Function MakeDecoElement(pMarkerSym As IMarkerSymbol, _
                    dPos As Double) _
                    As ISimpleLineDecorationElement
    Set MakeDecoElement = New SimpleLineDecorationElement
    MakeDecoElement.PositionAsRatio = True
    MakeDecoElement.Rotate = True
    MakeDecoElement.AddPosition dPos
    MakeDecoElement.MarkerSymbol = pMarkerSym
End Function

Function MakeArrowLineSym(lLineRGB As Long, dWidth As Double, lArrowRGB As Long) _
                    As ICartographicLineSymbol
    Dim pLineProps As ILineProperties
    Set pLineProps = New CartographicLineSymbol
    Set pLineProps.LineDecoration = New LineDecoration

    Dim pAMS As IArrowMarkerSymbol
    Set pAMS = New ArrowMarkerSymbol
    pAMS.Length = 4 * dWidth
    pAMS.size = 5 * dWidth
    pAMS.Width = 5 * dWidth
    pAMS.Color = MakeColor(lArrowRGB)
    pLineProps.LineDecoration.AddElement MakeDecoElement(pAMS, 1)

    Dim pLineSym As ILineSymbol
    Set pLineSym = pLineProps
    pLineSym.Color = MakeColor(lLineRGB)
```

```
    pLineSym.Width = dWidth

    Set MakeArrowLineSym = pLineSym
End Function

Public Sub UnionizeSegments(lineLayer As ILayer)
    Dim Response
    'Response = MsgBox("Sub UnionizeSegments", vbInformation)

    Dim pMxDocument As IMxDocument
    Set pMxDocument = ThisDocument
    Dim pID As New UID
    pID = "esriEditor.editor"
    Dim pApp As IApplication
    Set pApp = Application
    Dim pEditor As IEditor
    Set pEditor = pApp.FindExtensionByCLSID(pID)
    Dim pDataset As IDataset
    Set pDataset = lineLayer
    Dim pWorkspace As IWorkspace
    Set pWorkspace = pDataset.Workspace
    Dim pGeoCollection As IGeometryCollection
    Dim inFeatures As Integer
    Dim inParts As Integer
    Dim mergeSuccess As Boolean
    mergeSuccess = False

    pEditor.StartEditing pWorkspace

    Call SelectAll

    Dim pEnumFeat As IEnumFeature
    Set pEnumFeat = pEditor.EditSelection
    Dim pFeature As IFeature
    Set pFeature = pEnumFeat.Next
    Set pGeoCollection = pFeature.Shape
    inFeatures = pEditor.SelectionCount
    inParts = pGeoCollection.GeometryCount

    If pEditor.SelectionCount < 1 Then
        Response = MsgBox("ERROR: No features selected", vbCritical)
        Exit Sub
    End If
    If pEditor.SelectionCount > 1 Then
        Dim pUID As New UID
        Dim pCmdItem As ICommandItem
        pUID.Value = "esriEditor.MergeCommand"
        Set pCmdItem = Application.Document.CommandBars.Find(pUID)
        pCmdItem.Execute
```

```
    End If
    If pEditor.SelectionCount = 1 Then
      Set pEnumFeat = pEditor.EditSelection
      Set pFeature = pEnumFeat.Next
      Set pGeoCollection = pFeature.Shape
      Dim numParts As Integer
      numParts = pGeoCollection.GeometryCount
      If numParts > 1 Then
        'MsgBox ("Geometry count: " & pGeoCollection.GeometryCount)
        Response = MsgBox("Unionize failed! " & inFeatures & " line features merged into 1 line feature
and " & inParts & " line parts merged into " & pGeoCollection.GeometryCount & " line parts", vbCritical)
        pEditor.StopEditing (True)
        Exit Sub
        End
      Else
        Response = MsgBox("Success: " & inFeatures & " line features merged into " &
pEditor.SelectionCount & " line feature", vbInformation)
        pEditor.StopEditing (True)
      End If
    Else
      Response = MsgBox("Unionize failed! " & inFeatures & " line features merged into " &
pEditor.SelectionCount & " line features", vbCritical)
      pEditor.StopEditing (False)
      Exit Sub
      End
    End If

End Sub

Public Sub DeleteShapeFile(strPath As String, strShapefile As String)
    Dim pWSF As IWorkspaceFactory
    Dim pFWS As IFeatureWorkspace
    Dim pFC As IFeatureClass
    Dim pDS As IDataset
    Dim pPS As IPropertySet

    Set pPS = New PropertySet
    pPS.SetProperty "DATABASE", strPath

    Set pWSF = New ShapefileWorkspaceFactory
    Set pFWS = pWSF.Open(pPS, 0)

    Set pFC = pFWS.OpenFeatureClass(strShapefile)

    Set pDS = pFC
    pDS.Delete
End Sub

'Check if layer is editable
```

```
Public Function CheckEdit(theLayer As ILayer) As Boolean
    Dim pFLayer As IFeatureLayer
    Dim pFClass As IFeatureClass
    Dim pDataSetEditInfo As IDatasetEditInfo

    Set pFLayer = theLayer
    Set pFClass = pFLayer.FeatureClass
    Set pDataSetEditInfo = pFClass

    If pDataSetEditInfo.CanEdit = True Then
        CheckEdit = True
    Else
        CheckEdit = False
    End If

    Exit Function

ErrorHandler:
        MsgBox "An error has occured within CheckEdit." & vbCr & vbCr & _
            "Error Details : " & Err.Description, vbExclamation + vbOKOnly, "Error"

End Function
```

Form 1a

```
Private m_pMxDoc As IMxDocument
Private m_pMaps As IMaps

Private m_pMap As IMap                  'Pointer to a map in the maps collection
Private m_pEnumLayers As IEnumLayer     'Enumeration of layers in a map
Private m_pLayer As ILayer              'Pointer to a layer in a map

Private m_pFeatcls As IFeatureClass
Private m_pFeatLayer As IFeatureLayer

Private Sub cboMapLayers_Change()
    cmdOK.Enabled = True
End Sub

Private Sub cmdCancel_Click()
    Unload frm1A
End Sub

Private Sub cmdHelp_Click()
    If frm1A.Width < 250 Then
        frm1A.Width = 400
        cmdHelp.Caption = "<< Hide Help"
        cmdHelp.ControlTipText = "Hide Help"
```

```
   Else
      frm1A.Width = 225
      cmdHelp.Caption = "Show Help >>"
      cmdHelp.ControlTipText = "Show Help"
   End If
End Sub

Private Sub cmdOK_Click()
   Dim Response
   Dim numFeatures As Integer
   Dim numParts As Integer
   numFeatures = 0
   numParts = 0

   If IsNull(cboMapLayers.Value) Then
      MsgBox "Nothing selected"
   Else
      Set m_pMap = m_pMxDoc.FocusMap
      Set m_pEnumLayers = m_pMap.Layers
      Set m_pLayer = m_pEnumLayers.Next
      Do Until m_pLayer Is Nothing
         'MsgBox (m_pLayer.Name & "  " & cboMapLayers.Value)
         If cboMapLayers.Value = m_pLayer.Name Then
            frm1A.Hide
            Set g_pStrmLayer = m_pLayer
            Set m_pFeatLayer = g_pStrmLayer
            Set m_pFeatcls = m_pFeatLayer.FeatureClass
            ' check whether the stream centerline layer can be opened for editing
            If Not CheckEdit(m_pLayer) Then
               Response = MsgBox("ERROR: The selected stream centerline layer is not editable",
vbCritical)
               Unload frm1A
               End
            End If
            ' check for number of segments
            numFeatures = m_pFeatcls.FeatureCount(Nothing)
            ' check for empty shapefile
            If numFeatures = 0 Then
               Response = MsgBox("ERROR: The selected stream centerline layer has no features",
vbCritical)
               Unload frm1A
               End
            End If
            ' check the stream centerline's spatial reference
            Dim pSpatialRef As ISpatialReference
            Dim pGeoDataset As IGeoDataset
            Set pGeoDataset = m_pFeatLayer
            Set pSpatialRef = pGeoDataset.SpatialReference
            MsgBox "Stream centerline SR: " & pSpatialRef.Name
```

```
        If pSpatialRef.Name = "Unknown" Then
          Response = MsgBox("ERROR: Stream centerline spatial reference is undefined", vbCritical)
          Unload frm1A
          End
        End If
        If Left(pSpatialRef.Name, 3) = "GCS" Then
          Response = MsgBox("ERROR: Stream centerline spatial reference is geographic", vbCritical)
          Unload frm1A
          End
        End If
        ' check map units of shapefile spatial reference
        Dim pPCS As IProjectedCoordinateSystem
        Set pPCS = pGeoDataset.SpatialReference
        'MsgBox pPCS.CoordinateUnit.Name
        If pPCS.CoordinateUnit.Name = "Meter" Then
          g_LinearConverson = 1#
        ElseIf Left(pPCS.CoordinateUnit.Name, 4) = "Foot" Then
          g_LinearConverson = 3.2808399
        Else
          Response = MsgBox("ERROR: Unsupported spatial reference linear units: " &
pPCS.CoordinateUnit.Name, vbCritical)
          Unload frm1A
          End
        End If
        ' make selected stream centerline visible
        g_pStrmLayer.Visible = True
        ' assign the stream centerline renderer with arrow in direction of stream flow
        Call AssignRenderer
        ' update the view extent centered around the stream centerline
        m_pMxDoc.ActiveView.Extent = g_pStrmLayer.AreaOfInterest
        Dim pEnvelope As IEnvelope
        Set pEnvelope = m_pMxDoc.ActiveView.Extent
        pEnvelope.Expand 1.5, 1.5, True
        m_pMxDoc.ActiveView.Extent = pEnvelope
        m_pMxDoc.ActiveView.Refresh
        If numFeatures > 1 Then
          Response = MsgBox("The stream centerline is composed of " & numFeatures & " segments.
Unionize segments now?", vbYesNo)
          If Response Then
            Call UnionizeSegments(g_pStrmLayer)
          End If
          numFeatures = m_pFeatcls.FeatureCount(Nothing)
          If numFeatures > 1 Then
            Unload frm1A
            End
          End If
        End If
        If numFeatures = 1 Then
          ' check number of parts
```

```
            Dim pFeature As IFeature
            Set pFeature = m_pFeatcls.GetFeature(0)
            Dim pGeoCollection As IGeometryCollection
            Set pGeoCollection = pFeature.Shape
            numParts = pGeoCollection.GeometryCount
            If numParts > 1 Then
               Response = MsgBox("Profile is composed of 1 feature with " & numParts & " parts. Union
the parts now?", vbYesNo)
               If Response Then
                  Call UnionizeSegments(g_pStrmLayer)
               End If
            End If
            Set pFeature = m_pFeatcls.GetFeature(0)
            Set pGeoCollection = pFeature.Shape
            numParts = pGeoCollection.GeometryCount
            If numParts > 1 Then
               Unload frm1A
               End
            End If
         End If
         Response = MsgBox("Profile is continuous line", vbInformation)
         ' check the stream flow direction
         Response = MsgBox("Is this the direction of flow?", vbYesNo)
         If Response = vbNo Then
            'MsgBox ("Calling flip sub")
            Call ICurve_ReverseOrientation(True)
         Else
            Call ICurve_ReverseOrientation(False)
         End If
         ' if ICurve_ReverseOrientation failed, the stream length is set to -1
         If g_StreamLength = -1 Then
            Response = MsgBox("ERROR: The selected stream centerline layer is not editable",
vbCritical)
            Unload frm1A
            End
         End If
         'MsgBox ("here i am")
         Exit Do
      End If
      Set m_pLayer = m_pEnumLayers.Next
   Loop
   Call Step1B_Part1
   Unload frm1A
   ' hides the form
   Dim OpenForms
   OpenForms = DoEvents
 End If
End Sub
```

```
Private Sub imgHelp_Click()
End Sub

Private Sub UserForm_Initialize()
   Dim Response
   Set m_pMxDoc = ThisDocument
   Set m_pMaps = m_pMxDoc.Maps

   frm1A.Width = 225
   cmdOK.Enabled = False

   Set m_pMap = m_pMxDoc.FocusMap
   'MsgBox ("# of layers: " & m_pMap.LayerCount)

   If m_pMap.LayerCount > 0 Then
      Set m_pEnumLayers = m_pMap.Layers
      Set m_pLayer = m_pEnumLayers.Next

      cboMapLayers.Clear

      Do Until m_pLayer Is Nothing
         If TypeOf m_pLayer Is IFeatureLayer Then
            Set m_pFeatLayer = m_pLayer
            If m_pFeatLayer.valid Then
               Set m_pFeatcls = m_pFeatLayer.FeatureClass
               If m_pFeatcls.ShapeType = esriGeometryPolyline Then      'only place polyline features in
listbox
                  cboMapLayers.AddItem m_pLayer.Name
               End If
            End If
         End If
         Set m_pLayer = m_pEnumLayers.Next
      Loop
      'cboMapLayers.Text = cboMapLayers.List(0)
   Else
      Response = MsgBox("ERROR: No layers exist in the project", vbCritical)
      Unload frm1A
      End
   End If

End Sub
```

Form 1b:

```
Private Sub cmdCancel_Click()
      'frm1B.Hide
      Unload frm1B
End Sub
```

```
Private Sub cmdHelp_Click()
    If frm1B.Width < 250 Then
        frm1B.Width = 400
        cmdHelp.Caption = "<< Hide Help"
        cmdHelp.ControlTipText = "Hide Help"
    Else
        frm1B.Width = 225
        cmdHelp.Caption = "Show Help >>"
        cmdHelp.ControlTipText = "Show Help"
    End If
End Sub

Private Sub cmdOK_Click()
    Dim Response
    Dim valid As Boolean
    valid = False

    If FeatureExists(tbxShpFileName.Text) Then
        Response = MsgBox("ERROR: Shapefile " & tbxShpFileName.Text & " already exists", vbCritical)
        Exit Sub
    End If
    g_PntShpFlName = tbxShpFileName.Text

    frm1B.Hide
    Call Step1B_Part2
End Sub

Private Sub cmdShpFileBrowse_Click()
    Call OpenGxDialog
End Sub

Private Sub imgHelp_Click()
    If frm1B.Width < 250 Then
        frm1B.Width = 400
        imgHelp.ControlTipText = "Hide Help"
    Else
        frm1B.Width = 225
        imgHelp.ControlTipText = "Show Help"
    End If
End Sub

Private Sub lblFileName_Click()

End Sub

Private Sub lblNumPoints_Click()

End Sub
```

```
Private Sub tboDistance_AfterUpdate()
   Dim validNumeric As Boolean
   Dim validPositive As Boolean
   validNumeric = False
   validPositive = False
   If Not IsNull(tboDistance.Text) Then
      If IsNumeric(tboDistance.Text) Then
         tboDistance.Text = FormatNumber(tboDistance.Text, 0)
         validNumeric = True
         'MsgBox (tboDistance.Text)
         g_InputDistance = tboDistance.Text
         If g_InputDistance > 0 Then
            validPositive = True
         End If
      End If
   End If
   If Not validNumeric Then
      Response = MsgBox("ERROR: Invalid entry (distance value must be numeric)", vbCritical)
      Exit Sub
   End If
   If Not validPositive Then
      Response = MsgBox("ERROR: Invalid entry (distance value must be positive)", vbCritical)
      Exit Sub
   End If
   g_NumSegments = FormatNumber(g_StreamLength / g_InputDistance, 0)
   'If (g_StreamLength Mod g_InputDistance) > 0 Then
   '   g_NumSegments = g_NumSegments + 1
   'End If
   tboNumSegments.Text = FormatNumber(g_NumSegments, 0)
End Sub

Private Sub tboNumSegments_AfterUpdate()
   Dim valid As Boolean
   If Not IsNull(tboNumSegments.Text) Then
      If IsNumeric(tboNumSegments.Text) Then
         'MsgBox (tboDistance.Text)
         g_NumSegments = tboNumSegments.Text
         valid = True
      End If
   End If
   If Not valid Then
      Response = MsgBox("ERROR: Invalid entry (must be numeric)", vbCritical)
      Exit Sub
   End If
   g_InputDistance = g_StreamLength / g_NumSegments
   tboDistance.Text = FormatNumber(g_InputDistance, 0)
End Sub

Private Sub tbxShpFileName_Change()
```

```
  cmdOK.Enabled = False
  'MsgBox Dir(tbxShpFileName.Text)
  If Right(LCase(tbxShpFileName.Text), 4) = ".shp" Then
    Dim shpDir
    shpDir = SplitWorkspaceName(tbxShpFileName.Text)
    If Not IsNull(shpDir) Then
      Dim fs
      Set fs = CreateObject("Scripting.FileSystemObject")
      If fs.FolderExists(shpDir) Then
        cmdOK.Enabled = True
        'MsgBox "File Path: " & tbxShpFileName.Text
      End If
    End If
  End If
End Sub

Private Sub UserForm_Initialize()
  cmdOK.Enabled = False
  frm1B.Width = 225
  g_InputDistance = 25
  g_NumSegments = FormatNumber(g_StreamLength / g_InputDistance, 0)
  If (g_StreamLength Mod g_NumSegments) > 0 Then
    g_NumSegments = g_NumSegments + 1
  End If
  tboNumSegments.Text = FormatNumber(g_NumSegments, 0)
  lblStreamLength.Caption = "Stream Centerline Length (meters): " & FormatNumber(g_StreamLength,
2)
  tboDistance.Text = FormatNumber(g_InputDistance, 2)
End Sub
```

Form 2:

```
Private m_pMxDoc As IMxDocument
Private m_pMaps As IMaps

Private m_pMap As IMap                'Pointer to a map in the maps collection
Private m_pEnumLayers As IEnumLayer    'Enumeration of layers in a map
Private m_pLayer As ILayer            'Pointer to a layer in a map

Private m_pFeatcls As IFeatureClass
Private m_pFeatLayer As IFeatureLayer

Public g_pPointLayer As ILayer
Public g_pDEMLayer As ILayer


Private Sub cmdOK_Click()
  Dim Response
```

```
Dim numFeatures As Integer

If cboCenterlinePoint.Value = "" Then
   MsgBox "Please Select a Centerline Point Layer"
Else
If cboDEMLayer.Value = "" Then
   MsgBox "Please Select a DEM Layer"
Else
If cboDEMUnits.Value = "" Then
   MsgBox "Please Select a UNITS"
Else

   Set m_pMap = m_pMxDoc.FocusMap
   Set m_pEnumLayers = m_pMap.Layers
   Set m_pLayer = m_pEnumLayers.Next
   Do Until m_pLayer Is Nothing
      If cboCenterlinePoint.Value = m_pLayer.Name Then
          Set g_pPointLayer = m_pLayer
      End If

      If cboDEMLayer.Value = m_pLayer.Name Then
          Set g_pDEMLayer = m_pLayer
      End If

      Set m_pLayer = m_pEnumLayers.Next

   Loop

Dim demConvert As Double

If cboDEMUnits.Value = "Decimeters" Then
   g_DEMConvertUnits = 10
End If
If cboDEMUnits.Value = "Meters" Then
   g_DEMConvertUnits = 1
End If
If cboDEMUnits.Value = "Feet" Then
   g_DEMConvertUnits = 3.2808399
End If
If cboDEMUnits.Value = "Unknown Units of Great Interest" Then
   MsgBox "Very Funny...Please select other units, or contact your friendly neighborhood programmer"
Exit Sub
End If

frm2.Hide

' set mouse cursor to hourglass
Call BusyMouse(True)
```

```
    If obOneCell.Value = True Then
      'Here it is
      Call DEM_OnePointValue(g_pPointLayer, g_pDEMLayer, obNineCell.Value)
      'Call ProgDialog(g_pPointLayer, g_pDEMLayer, obNineCell.Value)

    Else

      ' add elevation fields
      ' Application.StatusBar.Message(0) = "Adding Elevation Fields"
      ' Call AddElevField(g_pPointLayer, obNineCell.Value)

      ' calculate the elevation at each point
      Application.StatusBar.Message(0) = "Calculating elevation"
      Call DEM_ValueTOPointFeatureClass(g_pPointLayer, g_pDEMLayer, obNineCell.Value)

      ' calculate the gradient at each point
      Application.StatusBar.Message(0) = "Calculating gradient"
      Call CalculateGradient(g_pPointLayer)

    End If


    ' set mouse cursor to default
    Call BusyMouse(False)
    Application.StatusBar.Message(0) = ""

    MsgBox "Grid Sampling Complete        ", vbInformation, "LS Step 3"

    Unload frm2

    End If
    End If
    End If
End Sub
Private Sub cmdCancel3_Click()
    Unload frm2
End Sub
Private Sub cmdHelp_Click()
    If frm2.Width < 250 Then
      frm2.Width = 420
      cmdHelp.Caption = "<< Hide Help"
      cmdHelp.ControlTipText = "Hide Help"
    Else
      frm2.Width = 220
      cmdHelp.Caption = "Show Help >>"
      cmdHelp.ControlTipText = "Show Help"
    End If
End Sub
```

```
Private Sub UserForm_Initialize()
  Dim Response
  Set m_pMxDoc = ThisDocument
  Set m_pMaps = m_pMxDoc.Maps

  Set m_pMap = m_pMxDoc.FocusMap

    frm2.Width = 220

  If m_pMap.LayerCount > 0 Then
    Set m_pEnumLayers = m_pMap.Layers
    Set m_pLayer = m_pEnumLayers.Next

    cboDEMLayer.Clear

    Do Until m_pLayer Is Nothing
      If m_pLayer.valid Then

        If TypeOf m_pLayer Is IFeatureLayer Then
          Set m_pFeatLayer = m_pLayer
          Set m_pFeatcls = m_pFeatLayer.FeatureClass
          If m_pFeatcls.ShapeType = esriGeometryPoint Then  'only place point features in combobox
            cboCenterlinePoint.AddItem m_pLayer.Name
          End If
        End If

        If TypeOf m_pLayer Is IRasterLayer Then
          Set m_pRLayer = m_pLayer
          cboDEMLayer.AddItem m_pLayer.Name
        End If

      End If
      Set m_pLayer = m_pEnumLayers.Next
    Loop

  Else
    Response = MsgBox("ERROR: No Raster layers exist in the project", vbCritical)
    Unload frmStep2
    End
  End If

  cboDEMUnits.AddItem "Decimeters"
  cboDEMUnits.AddItem "Meters"
  cboDEMUnits.AddItem "Feet"
  cboDEMUnits.AddItem "Unknown Units of Great Interest"

End Sub
```

Form 3:

```
Private m_pMxDoc As IMxDocument
Private m_pMaps As IMaps

Private m_pMap As IMap              'Pointer to a map in the maps collection
Private m_pEnumLayers As IEnumLayer    'Enumeration of layers in a map
Private m_pLayer As ILayer          'Pointer to a layer in a map

Private m_pFeatcls As IFeatureClass
Private m_pFeatLayer As IFeatureLayer

Private Sub cboContribLayer_Change()
   Set m_pMap = m_pMxDoc.FocusMap
   Set m_pEnumLayers = m_pMap.Layers
   Set m_pLayer = m_pEnumLayers.Next
   Do Until m_pLayer Is Nothing
      If cboContribLayer.Value = m_pLayer.Name Then
         Set g_pContribLayer = m_pLayer
      End If
      Set m_pLayer = m_pEnumLayers.Next
   Loop
   Verify_Inputs
End Sub

Private Sub cboReceivLayer_Change()
   Set m_pMap = m_pMxDoc.FocusMap
   Set m_pEnumLayers = m_pMap.Layers
   Set m_pLayer = m_pEnumLayers.Next
   Do Until m_pLayer Is Nothing
      If cboReceivLayer.Value = m_pLayer.Name Then
         Set g_pReceivLayer = m_pLayer
      End If
      Set m_pLayer = m_pEnumLayers.Next
   Loop
   Verify_Inputs
End Sub

Private Sub cmdCancel_Click()
   Unload frm3
End Sub

Private Sub cmdHelp_Click()
   If frm3.Width < 250 Then
      frm3.Width = 400
      cmdHelp.Caption = "<< Hide Help"
      cmdHelp.ControlTipText = "Hide Help"
   Else
      frm3.Width = 225
      cmdHelp.Caption = "Show Help >>"
```

```
      cmdHelp.ControlTipText = "Show Help"
   End If
End Sub

Private Sub cmdOK_Click()
   Dim Response

   ' did user select a contributing point layer
   If cboContribLayer.Value = "" Then
      cmdOK.Enabled = False
      Response = MsgBox("Please select a contributing point layer", vbCritical, "Step 6")
      Exit Sub
   End If
   ' did user select a receiving point layer
   If cboReceivLayer.Value = "" Then
      cmdOK.Enabled = False
      Response = MsgBox("Please select a receiving point layer", vbCritical, "Step 6")
      Exit Sub
   End If
   ' are the Contributing and Receiving layers the same layer
   If cboContribLayer.Value = cboReceivLayer.Value Then
      cmdOK.Enabled = False
      Response = MsgBox("The Contributing Layer and Receiving Layer cannot be the same layer",
vbCritical, "Step 6")
      Exit Sub
   End If
   ' did user select an output layer
   ' does the output layer already exist?
   If FeatureExists(tbxShpFileName.Text) Then
      Response = MsgBox("ERROR: Shapefile " & tbxShpFileName.Text & " already exists", vbCritical)
      Exit Sub
   End If
   g_pOutputLayerName = tbxShpFileName.Text

   Set m_pMap = m_pMxDoc.FocusMap
   Set m_pEnumLayers = m_pMap.Layers
   Set m_pLayer = m_pEnumLayers.Next
   Do Until m_pLayer Is Nothing
      If cboContribLayer.Value = m_pLayer.Name Then
         Set g_pContribLayer = m_pLayer
      End If
      If cboReceivLayer.Value = m_pLayer.Name Then
         Set g_pReceivLayer = m_pLayer
      End If
      Set m_pLayer = m_pEnumLayers.Next
   Loop

   Call Step6Control
```

```
End Sub

Private Sub cmdShpFileBrowse_Click()
   Call OpenGxDialog_Step6
End Sub

Private Sub imgHelp_Click()
   If frm3.Width < 250 Then
      frm3.Width = 400
      imgHelp.ControlTipText = "Hide Help"
   Else
      frm3.Width = 225
      imgHelp.ControlTipText = "Show Help"
   End If
End Sub

Private Sub tbxShpFileName_Change()
   cmdOK.Enabled = False
   'MsgBox Dir(tbxShpFileName.Text)
   If Right(LCase(tbxShpFileName.Text), 4) = ".shp" Then
      Dim shpDir
      shpDir = SplitWorkspaceName(tbxShpFileName.Text)
      If Not IsNull(shpDir) Then
         Dim fs
         Set fs = CreateObject("Scripting.FileSystemObject")
         If fs.FolderExists(shpDir) Then
            cmdOK.Enabled = True
            'MsgBox "File Path: " & tbxShpFileName.Text
         End If
      End If
   End If
End Sub

Private Sub UserForm_Initialize()
   Dim Response
   Set m_pMxDoc = ThisDocument
   Set m_pMaps = m_pMxDoc.Maps
   Set m_pMap = m_pMxDoc.FocusMap
   frm3.Width = 225
   cmdOK.Enabled = False

   If m_pMap.LayerCount > 0 Then
      Set m_pEnumLayers = m_pMap.Layers
      Set m_pLayer = m_pEnumLayers.Next

      cboContribLayer.Clear
      cboReceivLayer.Clear

      Do Until m_pLayer Is Nothing
```

```
        If m_pLayer.valid Then

          If TypeOf m_pLayer Is IFeatureLayer Then
            Set m_pFeatLayer = m_pLayer
            Set m_pFeatcls = m_pFeatLayer.FeatureClass
             If m_pFeatcls.ShapeType = esriGeometryPoint Then  'only place point features in combobox
              cboContribLayer.AddItem m_pLayer.Name
              cboReceivLayer.AddItem m_pLayer.Name
            End If
          End If

        End If
        Set m_pLayer = m_pEnumLayers.Next
      Loop
    Else
      Response = MsgBox("ERROR: No layers exist in the project", vbCritical)
      Unload frm3
      End
    End If

End Sub

Private Sub Verify_Inputs()
    If cboContribLayer.Value <> "" And cboReceivLayer.Value <> "" And tbxShpFileName.Value <> ""
Then
        cmdOK.Enabled = True
    Else
        cmdOK.Enabled = False
    End If
End Sub
```

Step 1:

```
Sub Step1B_Part1()
    Load frm1B
    frm1B.Show
End Sub

Sub Step1B_Part2()
    Unload frm1B
    ' hides the form
    Dim OpenForms
    OpenForms = DoEvents
    ' create point shapefile
    If Not IsNull(g_PntShpFlName) Then
      Dim pntShpFileName As String
      Dim pntShpFilePath As String
      pntShpFileName = SplitFileName(g_PntShpFlName)
```

```
    pntShpFilePath = SplitWorkspaceName(g_PntShpFlName)
    Dim pFeatcls As IFeatureClass
    'MsgBox pntShpFilePath & "\" & pntShpFileName
    Set pFeatcls = CreateShapefile(pntShpFilePath, pntShpFileName)
    'assign point shapefile same spatial reference as stream centerline
    Dim pLnSpatialRef As ISpatialReference
    Dim pPtSpatialRef As ISpatialReference
    Dim pLnGeoDataset As IGeoDataset
    Dim pPtGeoDataset As IGeoDataset
    Dim pLnFeatureLayer As IFeatureLayer
    'Dim pPtFeatureLayer As IFeatureLayer
    Set pLnFeatureLayer = g_pStrmLayer
    Set pLnGeoDataset = pLnFeatureLayer
    Set pLnSpatialRef = pLnGeoDataset.SpatialReference
    'Set pPtFeatureLayer = g_pStrmLayer
    Set pPtGeoDataset = pFeatcls
    Dim pGeodatasetAlterSpatRef As IGeoDatasetSchemaEdit
    Set pGeodatasetAlterSpatRef = pPtGeoDataset
    pGeodatasetAlterSpatRef.AlterSpatialReference pLnSpatialRef
    'MsgBox "Point shapefile SR: " & pPtGeoDataset.SpatialReference.Name
    If Not pFeatcls Is Nothing Then
       ' add point shapefile to TOC
       Call AddShapeFile(pntShpFilePath, pntShpFileName)
    Else
       MsgBox ("ERROR: New Shapefile not created")
       End
    End If
Else
    MsgBox ("No shapefile selected")
    End
End If

' set mouse cursor to hourglass
Call BusyMouse(True)

' populate point layer by creating points along centerline at input distance
Application.StatusBar.Message(0) = "Creating points along the curve"
Call CreatePointsAlongCurve
' populate LATDD and LONGDD fields for each point in the point layer
Application.StatusBar.Message(0) = "Populating latitude/longitude values at each point"
Call PopulateDDFields
' calculate the aspect of the stream centerline at each point in the point layer
Application.StatusBar.Message(0) = "Calculating aspect at each point"
Call CalcAspect

' set mouse cursor to default
Call BusyMouse(False)
Application.StatusBar.Message(0) = ""
```

```
    MsgBox "Profile Definition Complete        ", vbInformation, "LS Step 1"
End Sub
Sub AssignRenderer()
    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument

    Dim pGFLayer As IGeoFeatureLayer
    Set pGFLayer = g_pStrmLayer

    Dim pSRenderer As ISimpleRenderer
    Set pSRenderer = New SimpleRenderer

    Set pSRenderer.Symbol = MakeArrowLineSym(vbBlue, 2, vbBlue)
    'Set m_OldRenderer = pGFLayer.Renderer

    Set pGFLayer.Renderer = pSRenderer
    pMxDoc.CurrentContentsView.Refresh pGFLayer
End Sub

Sub SelectAll()
    Dim pMxDocument As IMxDocument
    Dim pFeatureSelection As IFeatureSelection

    Set pMxDocument = ThisDocument
    Set pFeatureSelection = g_pStrmLayer
    pFeatureSelection.SelectFeatures Nothing, esriSelectionResultNew, False

    'pMxDocument.ActiveView.Refresh
End Sub

Sub ICurve_ReverseOrientation(flipIt As Boolean)
    On Error GoTo errorHandle

    Dim pMxDocument As IMxDocument
    Set pMxDocument = ThisDocument
    Dim pID As New UID
    pID = "esriEditor.editor"
    Dim pApp As IApplication
    Set pApp = Application
    Dim pEditor As IEditor
    Set pEditor = pApp.FindExtensionByCLSID(pID)
    Dim pDataset As IDataset
    Set pDataset = g_pStrmLayer
    Dim pWorkspace As IWorkspace
    Set pWorkspace = pDataset.Workspace

    Dim Response
    pEditor.StartEditing pWorkspace
```

```
  Call SelectAll
  If pEditor.SelectionCount < 1 Then
    Response = MsgBox("ERROR: No features selected", vbCritical)
    Exit Sub
  End If

  Dim ftrReversed As Boolean
  ftrReversed = False

  Dim pEnumFeat As IEnumFeature
  Set pEnumFeat = pEditor.EditSelection
  Dim pFeature As IFeature
  Dim pCurve As ICurve

  Set pFeature = pEnumFeat.Next

  While Not pFeature Is Nothing
    If pFeature.Shape.GeometryType = esriGeometryPolyline Then
      Set pCurve = pFeature.Shape
      ' get the line length from the curve object (need to use it later)
      ' must convert to meters
      g_StreamLength = pCurve.Length / g_LinearConverson
      'MsgBox "Stream Length: " & g_StreamLength
      If flipIt Then
        pCurve.ReverseOrientation
        pFeature.Store
        ftrReversed = True
      End If
    End If
    Set pFeature = pEnumFeat.Next
  Wend

  pEditor.StopEditing (True)

  If ftrReversed Then
    pMxDocument.ActiveView.Refresh
    Response = MsgBox("Stream centerline direction reversed", vbInformation)
  End If

Exit Sub

errorHandle:
  If Err.Number = -2147220987 Then
    MsgBox "Unable to start edit session.  Can not edit read-only data: " + pDataset.Name
  Else
    MsgBox "Error initializing editor."
  End If
  g_StreamLength = -1
```

```
End Sub

Function CreateShapefile(sPath As String, sName As String) As IFeatureClass ' Dont include .shp
extension

    ' Open the folder to contain the shapefile as a workspace
    Dim pFWS As IFeatureWorkspace
    Dim pWorkspaceFactory As IWorkspaceFactory
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pFWS = pWorkspaceFactory.OpenFromFile(sPath, 0)

    ' Set up a simple fields collection
    Dim pFields As IFields
    Dim pFieldsEdit As IFieldsEdit
    Set pFields = New Fields
    Set pFieldsEdit = pFields

    Dim pField As IField
    Dim pFieldEdit As IFieldEdit

    ' populate array with field properties

    Dim fldList(10, 2) As Variant

    'field of data organized for Heat Source input

    fldList(0, 0) = "LENGTH"
    fldList(0, 1) = esriFieldTypeDouble
    fldList(1, 0) = "LONGDD"
    fldList(1, 1) = esriFieldTypeDouble
    fldList(2, 0) = "LATDD"
    fldList(2, 1) = esriFieldTypeDouble
    fldList(3, 0) = "ELEVATION"
    fldList(3, 1) = esriFieldTypeDouble
    fldList(4, 0) = "CHAN_WID"
    fldList(4, 1) = esriFieldTypeSingle
    fldList(5, 0) = "ASPECT"
    fldList(5, 1) = esriFieldTypeSingle


    'misc fields of data

    fldList(6, 0) = "GRADIENT"
    fldList(6, 1) = esriFieldTypeDouble
    fldList(7, 0) = "DEM_M"
    fldList(7, 1) = esriFieldTypeDouble

    'bk - add the x and y coordinates of the points using native coordinate system units
```

```
    fldList(8, 0) = "X"
    fldList(8, 1) = esriFieldTypeDouble
    fldList(9, 0) = "Y"
    fldList(9, 1) = esriFieldTypeDouble



    'bk

    ' Make the shape field
    ' it will need a geometry definition, with a spatial reference
    Set pField = New Field
    Set pFieldEdit = pField
    pFieldEdit.Name = "Shape"
    pFieldEdit.Type = esriFieldTypeGeometry

    Dim pGeomDef As IGeometryDef
    Dim pGeomDefEdit As IGeometryDefEdit
    Set pGeomDef = New GeometryDef
    Set pGeomDefEdit = pGeomDef
    With pGeomDefEdit
      .GeometryType = esriGeometryPoint
      Set .SpatialReference = New UnknownCoordinateSystem
    End With
    Set pFieldEdit.GeometryDef = pGeomDef
    pFieldsEdit.AddField pField

    For i = LBound(fldList) To (UBound(fldList) - 1)
       Set pField = New Field
       Set pFieldEdit = pField
       With pFieldEdit
          .Length = 8
          .Name = fldList(i, 0)
          .Type = fldList(i, 1)
       End With
       pFieldsEdit.AddField pField
    Next

    ' Create the shapefile
    ' (some parameters apply to geodatabase options and can be defaulted as Nothing)
    Dim pFeatClass As IFeatureClass
    Set pFeatClass = pFWS.CreateFeatureClass(sName, pFields, Nothing, _
                                Nothing, esriFTSimple, "Shape", "")

    Set CreateShapefile = pFeatClass
End Function

Sub AddShapeFile(sPath As String, sName As String)
   ' adds a shapefile to the map document
```

```
    Dim pWorkspaceFactory As IWorkspaceFactory
    Dim pFeatureWorkspace As IFeatureWorkspace
    Dim pFeatureLayer As IFeatureLayer
    Dim pMxDocument As IMxDocument
    Dim pMap As IMap

    ' create a new ShapefileWorkspaceFactory object and open a shapefile folder
    Set pWorkspaceFactory = New ShapefileWorkspaceFactory
    Set pFeatureWorkspace = pWorkspaceFactory.OpenFromFile(sPath, 0)
    ' create a new FeatureLayer and assign a shapefile to it
    Set pFeatureLayer = New FeatureLayer
    Set pFeatureLayer.FeatureClass = pFeatureWorkspace.OpenFeatureClass(sName)
    pFeatureLayer.Name = pFeatureLayer.FeatureClass.AliasName
    ' add the FeatureLayer to the focus map
    Set pMxDocument = Application.Document
    Set pMap = pMxDocument.FocusMap
    pMap.AddLayer pFeatureLayer
    Set g_pPointLayer = pFeatureLayer
End Sub

Function GetPolyline(strmLyr As ILayer) As IPolyline
    ' returns the first polyline in the stream centerline layer
    Dim pFeatLayer As IFeatureLayer
    Dim pFeature As IFeature
    Dim pFeatSel As IFeatureSelection
    Dim pFeatSet As ISelectionSet
    Dim pFCursor As IFeatureCursor
    Set pFeatLayer = strmLyr
    Set pFeatSel = pFeatLayer
    Set pFeatSet = pFeatSel.SelectionSet
    pFeatSet.Search Nothing, False, pFCursor
    Set pFeature = pFCursor.NextFeature
    If Not pFeature Is Nothing Then
        If pFeature.Shape.GeometryType = esriGeometryPolyline Then
            Set GetPolyline = pFeature.Shape
            Exit Function
        End If
    End If
End Function

Sub CreatePointsAlongCurve()
    'Creates points at a set distance along any feature implementing ICurve
    '
    'Justin Johnson
    'January 23, 2004
    'justin.johnson@geog.utah.edu
    '
    'Obtains selected features from currently-selected Layer
    'Stores new points in point theme at top of TOC
```

```
Dim pMxDoc As IMxDocument
Dim pMap As IMap
Dim pInGeometry As IGeometry
Dim pInLayer As ILayer
Dim pInFLayer As IFeatureLayer
Dim pOutFLayer As IFeatureLayer
Dim pInFCursor As IFeatureCursor
Dim pOutFCursor As IFeatureCursor
Dim pOutFBuffer As IFeatureBuffer
Dim pInFClass As IFeatureClass
Dim pOutFClass As IFeatureClass
Dim pSelSet As ISelectionSet
Dim pFSelection As IFeatureSelection
Dim pInFeature As IFeature
Dim pCurve As ICurve
Dim pPointCollection As IPointCollection
Dim pConstructMultipoint As IConstructMultipoint

Set pMxDoc = ThisDocument
Set pMap = pMxDoc.FocusMap
Set pInLayer = g_pStrmLayer

If pInLayer Is Nothing Then  'Check if no input layer is selected
  MsgBox "Select a feature layer in the TOC", vbCritical, "Incompatible input layer"
  Exit Sub
End If

If TypeOf pInLayer Is IFeatureLayer Then  'check if selected layer is a feature layer
  'Set pInFLayer = pMxDoc.SelectedLayer  'set selected layer as input feature layer
  Set pInFLayer = pInLayer  'set selected layer as input feature layer
Else
  MsgBox "Select a feature layer in the TOC", vbCritical, "Incompatible input layer"
  Exit Sub
End If

Set pOutFLayer = g_pPointLayer        ' set top layer in TOC as output feature layer
Set pInFClass = pInFLayer.FeatureClass
Set pOutFClass = pOutFLayer.FeatureClass

If Not pOutFClass.ShapeType = esriGeometryPoint Then  'check if output layer is Point type
  MsgBox "Geometry type of output layer is not Point", vbCritical, "Incompatible Output Layer"
  Exit Sub
End If

'Get selected features, if any
Set pFSelection = pInFLayer
Set pSelSet = pFSelection.SelectionSet
```

```
'Prompt user for distance between points
Dim pPointDist As Integer
pPointDist = g_InputDistance
'convert to feet
pPointDist = pPointDist * g_LinearConverson
'MsgBox ("Distance: " & Str(pPointDist))

'Create an Insert cursor on output feature class
Set pOutFBuffer = pOutFClass.CreateFeatureBuffer
Set pOutFCursor = pOutFClass.Insert(True)

  Set pInFCursor = pInFClass.Search(Nothing, True)

Dim k As Long 'count the number of points created
k = 0

Set pInFeature = pInFCursor.NextFeature

Do While Not pInFeature Is Nothing

  Set pInGeometry = pInFeature.Shape
  Set pCurve = pInGeometry
  ' get the line length from the curve object (need to use it later)
  ' must convert to meters
  'g_StreamLength = pCurve.Length / g_LinearConverson
  'MsgBox "Stream Length: " & g_StreamLength
  ' create the points here
  Set pConstructMultipoint = New Multipoint
  ' this is the ArcObject that does all of the heavy lifting
  pConstructMultipoint.ConstructDivideLength pCurve, pPointDist

  ' put the points in a collection
  Set pPointCollection = pConstructMultipoint

  ' write the points to the shapefile
  Dim i As Long
  For i = 0 To pPointCollection.PointCount - 1
    Set pOutFBuffer.Shape = pPointCollection.Point(i)  'store the new geometry
    pOutFCursor.InsertFeature pOutFBuffer
    k = k + 1
  Next i

Set pInFeature = pInFCursor.NextFeature

Loop

pMxDoc.ActiveView.Refresh
'MsgBox k & " points created in " & pOutFLayer.Name, vbInformation, "LS Step1B"
```

```
End Sub

Sub PopulateDDFields()
  ' populate the LATDD and LONGDD fields in the point shapefile
  Dim pMxDoc As IMxDocument
  Dim pMap As IMap
  Dim pInLayer As ILayer
  Dim pInFLayer As IFeatureLayer
  Dim pInGeoDataset As IGeoDataset
  Dim pInFCursor As IFeatureCursor
  Dim pInFClass As IFeatureClass
  Dim pSelSet As ISelectionSet
  Dim pFSelection As IFeatureSelection
  Dim pInFeature As IFeature
  Dim pInPoint As IPoint
  Dim dDist As Double

  Set pMxDoc = ThisDocument
  Set pMap = pMxDoc.FocusMap
  Set pInLayer = g_pPointLayer
  Set pInFLayer = pInLayer
  Set pInGeoDataset = pInFLayer
  Set pInFClass = pInFLayer.FeatureClass
  dDist = 0

  'get the indexes of the LAT, LON, and LENGTH fields
  Dim pFields As IFields
  Set pFields = pInFClass.Fields
  Dim latField As Integer
  latField = pFields.FindFieldByAliasName("LATDD")
  Dim lonField As Integer
  lonField = pFields.FindFieldByAliasName("LONGDD")
  Dim lenField As Integer
  lenField = pFields.FindFieldByAliasName("LENGTH")
  Dim xfield As Integer
  xfield = pFields.FindFieldByAliasName("X")
  Dim yfield As Integer
  yfield = pFields.FindFieldByAliasName("Y")


  'Get selected features, if any
  Set pFSelection = pInFLayer
  Set pSelSet = pFSelection.SelectionSet
  Set pInFCursor = pInFClass.Search(Nothing, True)
  Set pInFeature = pInFCursor.NextFeature ' get first point
  ' loop through points
  Do While Not pInFeature Is Nothing
    Set pInPoint = pInFeature.Shape
    ' project the point
```

```
        Dim pOutPoint As IPoint
        Set pOutPoint = New Point
        Set pOutPoint = CalcPointLatLong(pInPoint, g_pPointLayer)


        ' BK - store the projected native coordinates

        pInFeature.Value(xfield) = pInPoint.X
        pInFeature.Value(yfield) = pInPoint.Y
        pInFeature.Store

        'BK

        ' store the projected coordinates
        pInFeature.Value(latField) = pOutPoint.Y
        pInFeature.Value(lonField) = pOutPoint.X
        pInFeature.Value(lenField) = dDist
        pInFeature.Store
        'MsgBox "LATDD: " & pInFeature.Value(latField) & "   LONGDD: " & pInFeature.Value(lonField)
        ' get next point
        dDist = dDist + g_InputDistance
        If dDist > g_StreamLength Then
           dDist = g_StreamLength
        End If
        Set pInFeature = pInFCursor.NextFeature
    Loop

    'MsgBox "Latitude/longitude values calculated for all points", vbInformation, "LS Step1B"
End Sub

Sub CalcAspect()
    Dim pMxDoc As IMxDocument
    Dim pMap As IMap
    Dim pLnLayer As ILayer
    Dim pLnFLayer As IFeatureLayer
    Dim pLnFClass As IFeatureClass
    Dim pLnFeature As IFeature
    Dim pLnFCursor As IFeatureCursor
    Dim pPtLayer As ILayer
    Dim pPtFLayer As IFeatureLayer
    Dim pPtFClass As IFeatureClass
    Dim pPtFeature As IFeature
    Dim pPtFCursor As IFeatureCursor
    Dim pPolyline As IPolyline
    Dim pPoint As IPoint
    Dim dDist As Double
    Dim dAngle As Double
    Dim numPts As Long
```

```
  Set pMxDoc = ThisDocument
  Set pMap = pMxDoc.FocusMap
  Set pLnLayer = g_pStrmLayer
  Set pLnFLayer = pLnLayer
  Set pLnFClass = pLnFLayer.FeatureClass
  Set pPtLayer = g_pPointLayer
  Set pPtFLayer = pPtLayer
  Set pPtFClass = pPtFLayer.FeatureClass
  'Set pPolyline = GetPolyline(pLayer)

  ' get the index of the ID field
  Dim pLnFields As IFields
  Set pLnFields = pLnFClass.Fields
  Dim idField As Integer
  idField = pLnFields.FindFieldByAliasName("Id")
  ' get the index of the ID field
  Dim pPtFields As IFields
  Set pPtFields = pPtFClass.Fields
  Dim angField As Integer
  angField = pPtFields.FindFieldByAliasName("ASPECT")
  Dim lenField As Integer
  lenField = pPtFields.FindFieldByAliasName("LENGTH")
  ' get number of points to control progress bar
  numPts = pPtFClass.FeatureCount(Nothing)
  'MsgBox "Num points: " & Str(numPts)
  dDist = 0

  ' set line cursor
  Set pLnFCursor = pLnFClass.Search(Nothing, True)
  Set pLnFeature = pLnFCursor.NextFeature ' get first line
  Set pPolyline = pLnFeature.Shape
  ' set point cursor
  Set pPtFCursor = pPtFClass.Search(Nothing, True)
  Set pPtFeature = pPtFCursor.NextFeature ' get first point
  ' loop through features
  Do While Not pPtFeature Is Nothing
    dDist = pPtFeature.Value(lenField) * g_LinearConverson
    dAngle = GetAngle(pPolyline, dDist)
    'MsgBox "Distance: " & Str(dDist) & "    Aspect: " & Str(dAngle)
    pPtFeature.Value(angField) = dAngle
    pPtFeature.Store
    Set pPtFeature = pPtFCursor.NextFeature
    Dim angleField As Integer

  Loop

  Dim Response
  'MsgBox "Aspect calculated for all points", vbInformation, "LS Step1B"
End Sub
```

Step 2:

```
Sub DEM_OnePointValue(pStrmCenterlinePoint As ILayer, pDEMLayer As ILayer, numFields As
Boolean)

   Dim pMxDoc As IMxDocument
   Set pMxDoc = ThisDocument

   Dim pFLayer As IFeatureLayer

'Stream Centerline Point Layer from frm2
   Set pFLayer = pStrmCenterlinePoint

   Dim pFCur As IFeatureCursor
   Set pFCur = pFLayer.FeatureClass.Update(Nothing, False)

   Dim pInRaster As IRasterLayer

'DEM Layer from frm2
   Set pInRaster = pDEMLayer
   'MsgBox pInRaster.Name

   Dim pRaster As IRaster
   Dim pProp As IRasterProps

   Set pRaster = pInRaster.Raster
   Set pProp = pRaster

   Dim pIdentify As IIdentify
   Set pIdentify = pInRaster

   Dim pIDArray As IArray
   Dim pRIDObj As IRasterIdentifyObj
   Dim i As Long
   Dim pPoint As IPoint
   Dim pFeature As IFeature

   Dim pNewPoint As IPoint
   Set pNewPoint = New Point

   Dim pInFeatureClass As IFeatureClass
   Set pInFeatureClass = pFLayer.FeatureClass

      'Looping through each point

   Dim NumOfRow As Integer
   NumOfRow = pInFeatureClass.FeatureCount(Nothing)
```

```
    Dim DemFieldIndex As Integer
    Dim ElevFieldIndex As Integer


Dim pProDlgFact As IProgressDialogFactory
Dim pStepPro As IStepProgressor
Dim pProDlg As IProgressDialog2
Dim pTrkCan As ITrackCancel
Dim boolCont As Boolean

Dim thisElevation As Double
Dim previousElevation As Double

' Create a CancelTracker
Set pTrkCan = New CancelTracker

' Create the ProgressDialog. This automatically displays the dialog
Set pProDlgFact = New ProgressDialogFactory
Set pProDlg = pProDlgFact.Create(pTrkCan, Application.hWnd)

' Set the properties of the ProgressDialog
pProDlg.CancelEnabled = True
pProDlg.Description = "Calculating elevation (meters) and gradient"
pProDlg.Title = "LS Step 3"
pProDlg.Animation = esriProgressGlobe 'esriDownloadFile

' Set the properties of the Step Progressor
Set pStepPro = pProDlg
pStepPro.MinRange = 0
pStepPro.MaxRange = NumOfRow
pStepPro.StepValue = 1
pStepPro.Message = "Working..."


        Dim prvElevation As Double
      Dim prvLength As Double

  For i = 0 To NumOfRow - 1

  'If i = 0 Then
  '    previousElevation = 50000
  'End If

  ElevFieldIndex = pFCur.FindField("ELEVATION")
  DemFieldIndex = pFCur.FindField("DEM_M")
  If DemFieldIndex < 0 Then Exit Sub
  If ElevFieldIndex < 0 Then Exit Sub
```

```
   'Get point
 Set pFeature = pInFeatureClass.GetFeature(i)
 Set pPoint = pFeature.Shape
 pNewPoint.X = pPoint.X
 pNewPoint.Y = pPoint.Y
   'Get RasterIdentifyObject on that point
 Set pIDArray = pIdentify.Identify(pNewPoint)


 If Not pIDArray Is Nothing Then
   Set pRIDObj = pIDArray.Element(0)
       'Get the value of the RasterIdentifyObject and add it to the field
     If pProp.PixelType Then
       If pRIDObj.Name <> "NoData" Then

     '------

      pFeature.Value(DemFieldIndex) = CDbl(pRIDObj.Name) / g_DEMConvertUnits
      pFeature.Value(ElevFieldIndex) = CDbl(pRIDObj.Name) / g_DEMConvertUnits


      'If thisElevation > previousElevation Then
      '   pFeature.Value(DemFieldIndex) = previousElevation
      'Else
      '   pFeature.Value(DemFieldIndex) = thisElevation
      'End If

      '------

      pFeature.Store

     'previousElevation = pFeature.Value(DemFieldIndex)

       End If
     End If
 End If


 'Dim prvElevation As Double
 'Dim prvLength As Double

 LenFieldIndex = pFCur.FindField("LENGTH")
 GradFieldIndex = pFCur.FindField("GRADIENT")

 If prvElevation <> 0 Then
    pFeature.Value(GradFieldIndex) = ((prvElevation - pFeature.Value(ElevFieldIndex)) /
(pFeature.Value(LenFieldIndex) - prvLength))
```

```
            pFeature.Store
        End If

        pFeature.Store

        prvElevation = pFeature.Value(ElevFieldIndex)
        prvLength = pFeature.Value(LenFieldIndex)

boolCont = True
pStepPro.Message = "Processing" & Str(i) & " of " & NumOfRow & " Records"
Application.StatusBar.Message(0) = "Calculating Elevation and Gradient"
'Check if the cancel button was pressed. If so, stop process
boolCont = pTrkCan.Continue
If Not boolCont Then
Exit For
End If



        Next i

    Exit Sub

' Done
Set pTrkCan = Nothing
Set pStepPro = Nothing
Set pProDlg = Nothing
End Sub


Sub DEM_ValueTOPointFeatureClass(pStrmCenterlinePoint As ILayer, pDEMLayer As ILayer,
numFields As Boolean)

    Dim pMxDoc As IMxDocument
    Set pMxDoc = ThisDocument

    Dim pFLayer As IFeatureLayer

'Stream Centerline Point Layer from frm2
    Set pFLayer = pStrmCenterlinePoint

    Dim pFCur As IFeatureCursor
    Set pFCur = pFLayer.FeatureClass.Update(Nothing, False)

    Dim ElevFieldIndex As Integer
    Dim DemFieldIndex As Integer

    Dim pInRaster As IRasterLayer
```

```
'DEM Layer from frm2
  Set pInRaster = pDEMLayer
  'MsgBox pInRaster.Name

  Dim pRaster As IRaster
  Dim pProp As IRasterProps

  Set pRaster = pInRaster.Raster
  Set pProp = pRaster

  Dim pIdentify As IIdentify
  Set pIdentify = pInRaster

  Dim pIDArray As IArray
  Dim pRIDObj As IRasterIdentifyObj
  Dim i As Long
  Dim pPoint As IPoint
  Dim pFeature As IFeature

  Dim pNewPoint As IPoint
  Set pNewPoint = New Point

  Dim pInFeatureClass As IFeatureClass
  Set pInFeatureClass = pFLayer.FeatureClass

    'Looping through each point

  Dim NumOfRow As Integer
  NumOfRow = pInFeatureClass.FeatureCount(Nothing)

  Dim newElev As Double
  newElev = 0

  Dim lowElev As Double
  lowElev = 6194

  Dim thelastelev As Double

Dim pProDlgFact As IProgressDialogFactory
Dim pStepPro As IStepProgressor
Dim pProDlg As IProgressDialog2
Dim pTrkCan As ITrackCancel
Dim boolCont As Boolean

' Create a CancelTracker
Set pTrkCan = New CancelTracker

' Create the ProgressDialog. This automatically displays the dialog
```

```
Set pProDlgFact = New ProgressDialogFactory
Set pProDlg = pProDlgFact.Create(pTrkCan, Application.hWnd)

' Set the properties of the ProgressDialog
pProDlg.CancelEnabled = True
pProDlg.Description = "Calculating elevation (meters)"
pProDlg.Title = "LS Step 3"
pProDlg.Animation = esriProgressGlobe 'esriDownloadFile

' Set the properties of the Step Progressor
Set pStepPro = pProDlg
pStepPro.MinRange = 0
pStepPro.MaxRange = NumOfRow
pStepPro.StepValue = 1
pStepPro.Message = "Working..."

  For i = 0 To NumOfRow - 1

    thelastelev = lowElev

  'Get elevation value at the point
    Set pFeature = pInFeatureClass.GetFeature(i)
    Set pPoint = pFeature.Shape

    pNewPoint.X = pPoint.X
    pNewPoint.Y = pPoint.Y

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

  'Here's the stuff for the other cells
      Dim pRProps As IRasterProps
      Set pRProps = pRaster

      Dim cX As Double
      cX = pRProps.MeanCellSize.X

      Dim cY As Double
      cY = pRProps.MeanCellSize.Y

  'Up
      pNewPoint.X = pPoint.X
      pNewPoint.Y = pPoint.Y + cY

      Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

  'Up Left
      pNewPoint.X = pPoint.X - cX
      pNewPoint.Y = pPoint.Y + cY
```

```
        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Left
        pNewPoint.X = pPoint.X - cX
        pNewPoint.Y = pPoint.Y

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Down Left
        pNewPoint.X = pPoint.X - cX
        pNewPoint.Y = pPoint.Y - cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Down
        pNewPoint.X = pPoint.X
        pNewPoint.Y = pPoint.Y - cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Down Right
        pNewPoint.X = pPoint.X + cX
        pNewPoint.Y = pPoint.Y - cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Right
        pNewPoint.X = pPoint.X + cX
        pNewPoint.Y = pPoint.Y

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

    'Up Right
        pNewPoint.X = pPoint.X + cX
        pNewPoint.Y = pPoint.Y + cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

If numFields = "False" Then

      'Up Up Right

          pNewPoint.X = pPoint.X + cX
          pNewPoint.Y = pPoint.Y + 2 * cY

          Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

      'Up Up
```

```
        pNewPoint.X = pPoint.X
        pNewPoint.Y = pPoint.Y + 2 * cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Up Up Left

        pNewPoint.X = pPoint.X - cX
        pNewPoint.Y = pPoint.Y + 2 * cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Up Up Left Left

        pNewPoint.X = pPoint.X - 2 * cX
        pNewPoint.Y = pPoint.Y + 2 * cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Up Left Left

        pNewPoint.X = pPoint.X - 2 * cX
        pNewPoint.Y = pPoint.Y + cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Left Left

        pNewPoint.X = pPoint.X - 2 * cX
        pNewPoint.Y = pPoint.Y

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)
'Down Left Left

        pNewPoint.X = pPoint.X - 2 * cX
        pNewPoint.Y = pPoint.Y - cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Down Left Left

        pNewPoint.X = pPoint.X - 2 * cX
        pNewPoint.Y = pPoint.Y - 2 * cY

        Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Down Left

        pNewPoint.X = pPoint.X - cX
```

```
    pNewPoint.Y = pPoint.Y - 2 * cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Down

    pNewPoint.X = pPoint.X
    pNewPoint.Y = pPoint.Y - 2 * cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Down Right

    pNewPoint.X = pPoint.X + cX
    pNewPoint.Y = pPoint.Y - 2 * cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Down Right Right

    pNewPoint.X = pPoint.X + 2 * cX
    pNewPoint.Y = pPoint.Y - 2 * cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Down Right Right

    pNewPoint.X = pPoint.X + 2 * cX
    pNewPoint.Y = pPoint.Y - cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Right Right

    pNewPoint.X = pPoint.X + 2 * cX
    pNewPoint.Y = pPoint.Y

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Up Right Right

    pNewPoint.X = pPoint.X + 2 * cX
    pNewPoint.Y = pPoint.Y + cY

    Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

'Up Up Right Right

    pNewPoint.X = pPoint.X + 2 * cX
```

```
            pNewPoint.Y = pPoint.Y + 2 * cY

            Call ReturnElevation(pIDArray, pIdentify, pRIDObj, pNewPoint, pProp, newElev, lowElev)

        End If

              ElevFieldIndex = pFCur.FindField("ELEVATION")
              DemFieldIndex = pFCur.FindField("DEM_M")
              If DemFieldIndex < 0 Then Exit Sub
              If ElevFieldIndex < 0 Then Exit Sub

              If lowElev <= thelastelev Then

              pFeature.Value(DemFieldIndex) = lowElev
              pFeature.Value(ElevFieldIndex) = lowElev
              pFeature.Store

              Else

              pFeature.Value(DemFieldIndex) = thelastelev
              pFeature.Value(ElevFieldIndex) = thelastelev
              pFeature.Store

              End If


boolCont = True
pStepPro.Message = "Processing" & Str(i) & " of " & NumOfRow & " Records"
Application.StatusBar.Message(0) = "Calculating Elevation"
'Check if the cancel button was pressed. If so, stop process
boolCont = pTrkCan.Continue
If Not boolCont Then
Exit For
End If


     Next i

  Exit Sub
' Done
Set pTrkCan = Nothing
Set pStepPro = Nothing
Set pProDlg = Nothing
End Sub


Public Sub CalculateGradient(pStreamCenterlinePoint As ILayer)
  Dim pMxDoc As IMxDocument
```

```
    Set pMxDoc = ThisDocument

    Dim pFLayer As IFeatureLayer

'Stream Centerline Point Layer from frm2
    Set pFLayer = pStreamCenterlinePoint
    'Set pFLayer = pMxDoc.SelectedLayer

    Dim pFCur As IFeatureCursor
    Set pFCur = pFLayer.FeatureClass.Update(Nothing, False)

    Dim FieldIndex As Integer
    Dim DemFieldIndex As Integer
    Dim ElevFieldIndex As Integer
    Dim GradFieldIndex As Integer
    Dim LenFieldIndex As Integer

    Dim pPoint As IPoint
    Dim pFeature As IFeature

    Dim pInFeatureClass As IFeatureClass
    Set pInFeatureClass = pFLayer.FeatureClass

        'Looping through each point

    Dim NumOfRow As Integer
    NumOfRow = pInFeatureClass.FeatureCount(Nothing)

Dim pProDlgFact As IProgressDialogFactory
Dim pStepPro As IStepProgressor
Dim pProDlg As IProgressDialog2
Dim pTrkCan As ITrackCancel
Dim boolCont As Boolean

' Create a CancelTracker
Set pTrkCan = New CancelTracker

' Create the ProgressDialog. This automatically displays the dialog
Set pProDlgFact = New ProgressDialogFactory
Set pProDlg = pProDlgFact.Create(pTrkCan, Application.hWnd)

' Set the properties of the ProgressDialog
pProDlg.CancelEnabled = True
pProDlg.Description = "Calculating gradient"
pProDlg.Title = "TTools Step 3"
pProDlg.Animation = esriProgressGlobe 'esriDownloadFile

' Set the properties of the Step Progressor
Set pStepPro = pProDlg
```

```
pStepPro.MinRange = 0
pStepPro.MaxRange = NumOfRow
pStepPro.StepValue = 1
pStepPro.Message = "Thank you for your continued patience"


  For i = 0 To NumOfRow - 1


     Set pFeature = pInFeatureClass.GetFeature(i)
     Set pPoint = pFeature.Shape


     Dim prvElevation As Double
     Dim prvLength As Double

     LenFieldIndex = pFCur.FindField("LENGTH")
     GradFieldIndex = pFCur.FindField("GRADIENT")
     ElevFieldIndex = pFCur.FindField("ELEVATION")

     If prvElevation <> 0 Then
        pFeature.Value(GradFieldIndex) = ((prvElevation - pFeature.Value(ElevFieldIndex)) /
(pFeature.Value(LenFieldIndex) - prvLength))
        pFeature.Store

     End If

     pFeature.Store

     prvElevation = pFeature.Value(ElevFieldIndex)
     prvLength = pFeature.Value(LenFieldIndex)


boolCont = True
pStepPro.Message = "Processing" & Str(i) & " of " & NumOfRow & " Records"
Application.StatusBar.Message(0) = "Calculating Gradient"
'Check if the cancel button was pressed. If so, stop process
boolCont = pTrkCan.Continue
If Not boolCont Then
Exit For
End If


     Next i

  Exit Sub
' Done
Set pTrkCan = Nothing
Set pStepPro = Nothing
```

```
Set pProDlg = Nothing

End Sub

Sub ReturnElevation(pIDArray As IArray, pIdentify As IIdentify, pRIDObj As IRasterIdentifyObj,
pNewPoint As IPoint, pProp As IRasterProps, newElev As Double, lowElev As Double)

        Set pIDArray = pIdentify.Identify(pNewPoint)
        If Not pIDArray Is Nothing Then
        Set pRIDObj = pIDArray.Element(0)
           'Get the value of the RasterIdentifyObject and add it to the field
           If pProp.PixelType Then
              If pRIDObj.Name <> "NoData" Then
                 newElev = CDbl(pRIDObj.Name) / g_DEMConvertUnits

                 If newElev < lowElev Then
                    lowElev = newElev
                 End If
              End If
           End If
        End If

End Sub
```

Step 3:

```
Private m_pMxDoc As IMxDocument
Private m_pMaps As IMaps
Private m_pMap As IMap              'Pointer to a map in the maps collection

Public Sub Step6Control()
   Unload frm3

   Dim tmpName As IName
   Dim pOutPutName As IName
   Dim pOutDataSetName As IDatasetName
   Dim pOutWorkspaceName As IWorkspaceName
   Dim pOutFeatClsName As IFeatureClassName
   Dim pWorkspace As IWorkspace
   '++ Create the target workspace for the output dataset
   Set pOutWorkspaceName = New WorkspaceName
   pOutWorkspaceName.WorkspaceFactoryProgID = "esriCore.ShapefileWorkspaceFactory.1"
   pOutWorkspaceName.PathName = SplitWorkspaceName(g_pOutputLayerName)
   Set pOutFeatClsName = New FeatureClassName
   Set pOutDataSetName = pOutFeatClsName
   Set pOutDataSetName.WorkspaceName = pOutWorkspaceName
   Set tmpName = pOutWorkspaceName
   Set pWorkspace = tmpName.Open
   '++ set the shapefile entry name
```

```
   pOutDataSetName.Name = SplitFileName(g_pOutputLayerName)
   Set pOutPutName = pOutDataSetName

   Dim joinLayer As IFeatureLayer
   Set joinLayer = JoinByLocation(g_pContribLayer, g_pReceivLayer, pOutPutName)

End Sub
Public Sub JoinPoints()
   MsgBox "JoinByLocation(g_pContribLayer,g_pReceivLayer)", vbInformation
End Sub
Public Function JoinByLocation(ByVal pFeatLyrSrc As IFeatureLayer, ByVal pFeatLyrJn As
IFeatureLayer, ByVal pFeatLyrName As IName) As IFeatureLayer
   '++ Spatial Joins
   '++ Assumes feature layers (both shapefiles) share the same co-ordinate system.
   On Error GoTo ErrorHandler
   '++ Variable declaration
   Dim Response
   Dim pTableSrc As ITable
   Dim pTableJn As ITable
   Dim pFeatclsOut As IFeatureClass
   Dim pSpatialJoin As ISpatialJoin
   'Dim pAggOpts As IAggregateOptions
   Dim maxMapDist As Double
   '++ Create the source and join tables
   Set pTableSrc = pFeatLyrSrc
   Set pTableJn = pFeatLyrJn
   Set pSpatialJoin = New SpatialJoin
   '++ Set the properties of the SpatialJoin object
   With pSpatialJoin
      .ShowProcess(True) = 0
      .LeftOuterJoin = True
      Set .SourceTable = pTableSrc
      Set .JoinTable = pTableJn
   End With
   '++ Specify the value for maxMapDist parameter : -1 is the default (infinity)
   '++ Other negative numbers are invalid and will produce an empty output feature class
   maxMapDist = -1
   '++ QI for IAggregateOptions to include the max values
   '++ of numeric fields
   'Set pAggOpts = pSpatialJoin
   'pAggOpts.IsMax = True

   '++ Execute the spatial join using the JoinAggreate method
   'Set pFeatclsOut = pSpatialJoin.JoinAggregate(pOutPutName, maxMapDist)
   Set pFeatclsOut = pSpatialJoin.JoinNearest(pFeatLyrName, maxMapDist)
   If Not pFeatclsOut Is Nothing Then
      Dim pFeatLayerOut As IFeatureLayer
      Set pFeatLayerOut = New FeatureLayer
      Set pFeatLayerOut.FeatureClass = pFeatclsOut
```

```
      Set JoinByLocation = pFeatLayerOut
      ' add point shapefile to TOC
      Call AddShapeFile(SplitWorkspaceName(g_pOutputLayerName),
SplitFileName(g_pOutputLayerName))
   End If

   Set pSpatialJoin = Nothing
   Exit Function
ErrorHandler:
   'MsgBox "JoinByLocation_Aggregate", Err.Number, Err.descripion
   VBA.MsgBox "JoinByLocation", Err.Number, Err.descripion
End Function

Sub OpenGxDialog_Step6()
   Dim pGxdial As IGxDialog
   Set pGxdial = New GxDialog
   pGxdial.ButtonCaption = "OK"
   pGxdial.Title = "Create Join Output Point Shapefile"
   pGxdial.RememberLocation = True
   Dim pShapeFileObj As IGxObject

   Dim pGxFilter As IGxObjectFilter
   Set pGxFilter = New GxFilterShapefiles 'e.g shp
   Set pGxdial.ObjectFilter = pGxFilter

   If pGxdial.DoModalSave(ThisDocument.Parent.hWnd) Then
      Dim pLocation As IGxFile
      Dim fn As String
      Set pLocation = pGxdial.FinalLocation
      fn = pGxdial.Name
   End If
   If Not pLocation Is Nothing Then
      g_pOutputLayerName = pLocation.Path & "\" & fn
      frm3.tbxShpFileName.Text = g_pOutputLayerName
      frm3.cmdOK.Enabled = True
   End If
End Sub
```

REFERENCES

Bowen, Z.H., Waltermire, R.G. (2002). "Evaluation of Light Detection and Ranging (LiDAR) For Measuring River Corridor Topography." *Journal of the American Water Resources Association, 38*(1), 33-41.

Cavalli, M.C., Tarolli, P., Marchi, L., Fontana, G.D. (2007). "The effectiveness of airborne LiDAR data in the recognition of channel-bed morphology." *Catena*, doi:10.1016/j.catena

Charlton, M.E., Large, A.R.G., Fuller, I.C. (2003). "Application of airborne LiDAR in River Environments: The River Croquet, Northumberland, UK." *Earth Surface Processes and Landforms*, *28*, 299-306.

Gangodagamage, C., Barnes, E., Foufoula-Georgiou, E. (2007). "Scaling in river corridor widths depicts organization in valley morphology." *Geomorphology*, doi:10.1016/j.geomor.

Gilvear, D., Tyler, A., Davids, C. (2004). "Detection of estuarine and tidal river hydromorphology using hyperspectral and LiDAR data: Forth estuaary, Scotland." *Estuarine, Coastal and Shelf Science*, *61*, 379-392.

Glenn, N.F., Streutker, D.R., Chadwick, D.J., Thackray, G.D., Dorsch, S.J. (2006). "Analysis of LiDAR-derived topographic information for characterizing and differentiating landslide morphology and activity." *Geomorphology*, *73*, 131-148.

Halwas, Karen L., Church, Michael. (2001). "Channel units in small, high gradient streams on Vancouver Island, British Columbia." *Geomorphology*, *43*, 243-256

Harrelson, C.C., Rawlins, C.L., Potyondy, J.P. (1994). *Stream Channel Reference Sites: An Illustrated Guide to Field Technique*. Fort Collins, CO: USDA Forest Service, Rocky Mountian and Range Experiment Station.

Hilldale, R.C., Raff, D. (2008). "Assessing the ability of airborne LiDAR to map river bathymetry." *Earth Surface Processes and Landforms*, *33*, 773-783.

James, A. L., Watson, D.G., Hansen, W.F. (2007). "Using LiDAR data to map gullies and headwater streams under forest canopy: South Carolina, USA." *Catena*, *71*, 132-144.

Jenkins, Francis A., White, Harvey, E. (1957). *Fundamentals of Optics*. New York: McGraw-Hill Book Company.

Knighton, David A. (1999, January 1). "Downstream variation in stream power." *Geomorphology*, *29*, 293-306.

Leica Geoystems AG (2007). *Leica ALS50 II Airborne Laser Scanner Product Specifications.* Heerbrugg, Switzerland: Leica Geoystems AG.

Lin, Z., Oguchi, T. (2006, January 13). "DEM analysis on longitudinal and transverse profiles of steep mountainous watersheds." *Geomorphology*, *78*, 77-89.

Magirl, C.S., Webb R.H., Griffiths, P.G. (2005). "Changes in water surface profile of the Colorado River in Grand Canyon, Arizona, between 1923 and 2000." *Water Resources Journal*, *41*, W05021.doi:10.1029/2

Montgomery, D.R., Buffington, J.M. (1997, May). "Channel-reach morphology in mountian drainage basins." *GSA Bulletin*, *109(5),* 596-611.

Maslov, D.V., Fadeev, V.V., Lyashenko, A.I. (2000, June16th – 17th). "A Shoreline-based LiDAR for Coastal Seawater Monitoring." *Proceedings of EARSeL-SIG-Workshop LiDAR, Dresden/FRG.*

Reusser, L., Bierman, P. (2007). "Accuracy assessment of LiDAR-derived DEMs of bedrock river channels: Holtwood Gorge, Susquehanna River." *Geophysical Letters, 34*, L23S06

Schumann G., Matgen, P., Cutler, M.E.J., Black, A., Hoffman, L., Pfister, L. (2008). "Comparison of remotely sensed water stages from LiDAR, topographic contours and SRTM." *ISPRS Journal of Photogrammetry & Remote Sensing, 63*, 283-296.

Singh V.P., Zhang, L. (2006). "At-a-station hydraulic geometry relations, 1: theoretical development." *Hydrological Processes.* DOI: 10.1002/hyp.6411

Smith, L.C., Sheng, Y., Magilligan, F.J., Smith, N.D., Gomez, B., Mertez, L.A.K., Krabill, W.B., Garvin, J.B. (2006). "Geomorphic impact and rapid subsequent recovery from the 1996 Skeioararsandur jokulhlaup, Iceland, measured with multi-year airborne LiDAR." *Geomorphology, 75*, 65-75.

Soininen, Arttu. (2005) *TerraScan User's Guide*. Artuu Soininen, Terrasolid. All rights reserved.

Su, L. et al. (2007). "Simple LiDAR detecting wake profiles." *Journal of Optics A: Pure and Applied Optics, 9,* 842-847.

Thoma, D.P., Gupta, S.C., Bauer, M.E., Kirchoff, C.E. (2005). "Airborne laser scanning for riverbank erosion assessment." *Remote Sensing of Environment, 95*, 493-501.

U.S. Army Corps of Engineers (1993, October 15). *Chapter 6 Steady Flow-Water Surface Profiles: River Hydraulics*. Washington D.C., Department of the Army. EM1110-2-1416

Wang Chi-Kuei, Philpot, William D. (2007, August 4). "Using airborne bathymetric LiDAR to detect bottom type variation in shallow waters." *Remote Sensing of Environment, 106*, 123-135.

Western, A.W., Finlayson, B.L., McMahon, T.A., O'neill, I.C. (1997). "A method for characterizing logitudinal irregularity in river channels." *Geomorphology, 21*, 39-51.