Software Specifications for JXL Parser

Prepared by: Marshall Mattingly

Last updated: 9/10/2015

Table of Contents

1. Introduction

This section outlines the purpose, scope, and system overview of this document, as well as any definitions and references required for understanding this document.

1.1. Purpose

This document shall describe the requirements and specifications for the creation of a database containing all relevant information for the cataloging of imagery flights meant for image detection. This document shall also consider the requirements for a script to automate the parsing of data from flight record files (JXL format) and insertion into the described database. All data formats shall be enumerated with full descriptions, where possible.

1.2. Scope

This document is only concerned with the creation, implementation, and automated data entry of the database described in the purpose. The database creation and maintenance are out of the scope of this product. The database is intended to be used in corresponding systems, such as automated image recognition and web-based interfaces where users can manually detect objects. As such, any other database elements are out of the scope of this document, and shall be described in other documentation. Since the database can be queried by any software with access, no external connectors need to be defined.

1.3. Definitions

[To be updated with a full definitions list]

1.4. References

[To be updated with future system documents]

1.5. System Overview

This product is meant to take the raw data provided by the Trimble software during imaging flights and ingesting the data into a database for research purposes, such as image recognition, web-site presentation, or other representations. Therefore, this product is interested in making the data accessible, while the other products in the system are interested in presenting or manipulating the data.

2. General Description

This section gives a high-level overview of the integration of the software, functionality of the software, user interface characteristic of the software, constraints of the software, and assumptions and dependencies of the software.

2.1. Product Perspective

This software is related to a suite of products used for object detection in images, specifically of the identification of various animal species from aerial photos. This particular product describes the database used by all other products in the suite and defines software for the automated insertion of data into the database from the flight record information, which is stored in JXL format.

2.2. Product Functions

This product shall parse a given JXL file for image metadata and insert the data into the database described in this document.

2.3. User Characteristics

No user interface is intended to be created with this product. Users shall interact with this product via the command line.

2.4. Assumptions and Dependencies

The database shall be stored in a relational database on a database server with the format described in this document. Connection information to the database shall be provided by the user, and the information must be correct and the database server must be online for information to be inserted into the database.

3. Specific Requirements

This section covers the specific requirements the software must meet in order to be considered complete and functional. Use cases are utilized for clarity and specific objects, classes, database elements, and other designs, which must be implemented with specific data and functionality, are included for software design clarity.

3.1. External Interface Requirements

This section covers all requirements for external interactions, including interactions with the user, interactions with the hardware, interactions with other software, and communication with external sources.

3.1.1. User Interfaces

The user shall interact with this product via the command line, passing arguments for the database connection and file for parsing.

3.1.2. Hardware Interfaces

This product requires a network connection to connect to the database.

3.1.3. Software Interfaces

This product shall utilize existing database libraries connect and maintain a persistent connection to the database.

3.1.4. Communication Interfaces

This product shall connect to the database via the TCP/IP protocol, requiring a network connection to be available to the database server.

3.2. Functional Requirements

This section enumerates the functional requirements of the software. All functional requirements must be satisfied by the software to meet the needs of the project.

3.2.1. FREQ1 – Parse image and flight data from a given JXL file

3.2.1.1. Introduction

The software shall be able to parse all image and flight information from a JXL file, following the format listed in 3.4.1. Information for a single image is stored in three objects within the JXL file: ImageRecord, PhotoStationRecord, and PointRecord.

3.2.1.2. Inputs

A single JXL file, which shall be read by the software.

3.2.1.3. Processing

The parser shall process the entire JXL document tree, cross-referencing the ImageRecord, PhotoStationRecord, and PointRecord objects relating to a single image to gather all data for storage in the database.

3.2.1.4. Outputs

An array of Image objects for storage in the database.

3.2.1.5. Error Handling

Notifications shall be presented if the JXL file is unable to be read. Specifically, if the file is unable to be opened, the file is unable to be parsed, or no Image data is found within the JXL file.

3.2.2. FREQ2 – Connection to the database

3.2.2.1. Introduction

In order to store persistent data, the parser shall connect with a relational database. The user shall be able to define the connection information when the software is executed.

3.2.2.2. Inputs

The server name, user name, password, and database information for the connection to the relational database.

3.2.2.3. Processing

The inputs are used to connect to the database and create a persistent connection object.

3.2.2.4. Output

A persistent-connection object for connection to the database by other elements of the software.

3.2.2.5. Error Handling

Notifications shall be presented if the database connection fails, including any reasons returned by the database software. An error shall be thrown or an empty database object returned in the case of an error.

3.2.3. FREQ3 – Insert Image objects into the database

3.2.3.1. Introduction

The software shall be able to insert Image and Flight objects into the database. Any Image or Flight already found to be within the database shall be ignored.

3.2.3.2. Inputs

An array of Images to be inserted into the database.

3.2.3.3. Processing

Any Images provided, especially those created by 3.2.1, are first used to query the database. If the Image already exists in the database (all meta-data matches), then the Image is ignored. If the Image does not yet exist in the database, it is inserted.

3.2.3.4. Output

True if the Images was inserted or already exists in the database. False on error.

3.2.3.5. Error Handling

Notifications shall be presented if there is an error inserting the Image into the database. The connection is assumed to be working, as in 3.2.2.

3.3. Use Cases

This section diagrams and describes the use cases of this product in relation to all agents working with the software.

3.4. Classes / Objects / Data structures

This section describes all classes, object, and data structures that must be present in the software.

3.4.1. JXL data structure

The JXL data structure contains the flight and imagery information available for each flight. This section provides the entity name, entity nesting depth, data type, and description for all entities and data within the JXL data structure, as seen in Table 1. This structure and description is not to be considered definitive and shall be updated as new information becomes apparent.

Lvl	Entity	Type	Description

1	JOBFile {
	product,
	TimeStamp,
	jobName,
	version,
	productVersion,
	FieldBook
	}
		-	

2	PhotoInstrumentRecord {
	TimeStamp,
	ID
	}
		-	Information about the photo instrumentation. Appears to reference the UAV, not the camera.
			
			Timestamp is the time the instrument record is recorded.
			
			ID is the internal ID for the instrument record.

3	Type	String	Unknown. Appears to always be Aerial.
3	Model	String	Model of the photo instrument. Appears to always be Trimble UX5.
3	Serial	String	Serial number of the photo instrument.
3	FirmwareVersion	String	Firmware version of the photo instrument.
3	UserDefinedName	String	Unknown. Appears to always be Prototype.

2	CameraDesignRecord {
	TimeStamp,
	ID
	}
		-	Manufacturer and capability information related to a camera used to take images.
			
			Timestamp is the time the camera design was recorded.
			
			ID is the internal ID of the camera design.

3	Type	String	Model of the camera.
3	HeightPixels	Int	Source height of the camera images.
3	WidthPixels	Int	Source width of the camera images.
3	PixelSize	Float	Real size of each pixel of the camera. Extremely small.
3	LensModel	String	Type of lens used on the camera. Appears to always be Rectilinear.
3	NominalFocalLength	Float	Nominal focal length of the camera.

2	CameraRecord2 {
	TimeStamp,
	ID
	}
		-	Used throughout the JXL to reference a specific camera that was used to take images.
			
			Timestamp is the time the camera record was recorded.
			
			ID is the internal identifier for the camera record.

3	CameraDesignID	Int	ID of the CameraDesignRecord related to this camera.
3	CameraPosition	Int	Unknown. Appears to always be 01.
3	Optics	-	
4	IdealAngularMagnification	Float	Unknown. Appears to always by 1.0.
4	AngleSymmetricDistortion	-	
5	Order3	Float	Unknown.
5	Order5	Float	Unknown.
5	Order7	Float	Unknown.
5	Order9	Float	Unknown.
4	AngleDecenteringDistortion	-	
5	Column	Float	Unknown.
5	Row	Float	Unknown.
3	Geometry	-	
4	PerspectiveCenterPixels	-	
5	PrincipalPointColumn	Float	Unknown.
5	PrincipalPointRow	Float	Unknown.
5	PrincipalDistance	Float	Unknown.

4	VectorOffset	-	
5	X	Float	Unknown.
5	Y	Float	Unknown.
5	Z	Float	Unknown.
4	BiVectorAngle	-	
5	XX	Float	Unknown.
5	YY	Float	Unknown.
5	ZZ	Float	Unknown.

2	PointRecord {
	TimeStamp,
	ID
	}
			Location information for an image.
			
			Timestamp is the time the image was taken.
			
			ID is the internal identifier for this point record.

3	Name	String	The name of the image (without the .JPG extension).
3	Code		Unknown.
3	Method	String	Method for the location data. Appears to always be Coordinates.
3	SurveyMethod	String	Method for the survey. Appears to always be Autonomous.
3	Classification	String	Unknown. Appears to always be Normal.
3	Deleted	Bool	Unknown. Appears to always be false.
3	WGS84	-	Holds the location information. May dependent on the Method.
4	Latitude	Float	Latitude location.
4	Longitude	Float	Longitude location.
4	Height	Float	Height from which the image was recorded. See Environment for distance unit.

2	PhotoStationRecord {
	TimeStamp,
	ID
	}
			Holds information about the instrumentation and orientation for an image.
			
			Timestamp is the time the image was taken.
			
			ID is the internal identifier for the photo station record.

3	StationName	String	The name of the image (without the .JPG extension).
3	InstrumentHeight	0	Unknown. Appears to always be 0.

3	RawInstrumentHeight	-	
4	MeasurementMethod	String	Unknown. Appears to always be TrueHeight.
4	MeasuredHeight	0	Unknown. Appears to always be 0.
4	HorizontalOffset	0	Unknown. Appears to always be 0.
4	VerticalOffset	0	Unknown. Appears to always be 0.
3	InstrumentID	Int	ID of the instrument used. See: PhotoInstrumentRecord.
3	AtmosphereID	0	Unknown. Appears to always be 0.
3	StationType	String	Unknown. Appears to always be RawSensorValues.
3	DeviceAxisOrientationData	-	
4	DeviceAxisOrientation	-	
5	BiVector	-	
6	XX	Float	X-axis rotation during imaging. (roll)
6	YY	Float	Y-axis rotation during imaging. (pitch)
6	ZZ	Float	Z-axis rotation during imaging. (yaw)
4	MeasurementMethod	String	Unknown. Appears to always be TrueHeight.
4	MeasuredHeight	0	Unknown. Appears to always be 0.
4	HorizontalOffset	0	Unknown. Appears to always be 0.
4	VerticalOffset	0	Unknown. Appears to always be 0.

2	ImageRecord {
	TimeStamp,
	ID
	}
			Information about the image itself.
			
			Timestamp is the time the image was taken.
			
			ID is the internal identifier for the image record.

3	StationID	String	ID of the PhotoStationRecord for this image.
3	BackBearingID	-	Unknown. Appears to always be empty.
3	CameraID	Int	ID of the CameraRecord2 used to take this image.
3	PointRecordID	-	Unknown. Appears to always be empty.
3	FileName	String	Filename for the image, with .JPG extension.
3	HorizontalAngle	-	Unknown. Appears to always be empty.
3	VerticalAngle	-	Unknown. Appears to always be empty.
3	Width	Int	Width of the stored image.
3	Height	Int	Height of the stored image.
3	SourceX	Int	Unknown. Appears to always be 0.
3	SourceY	Int	Unknown. Appears to always be 0.

3	SourceWidth	Int	Width of the original device image.
3	SourceHeight	int	Height of the original device image.

2	AtmosphereRecord {
	TimeStamp,
	ID
	}
			Atmospheric information recorded by the device.
			
			Timestamp is the time that the atmospheric record was recorded.
			
			ID is the internal identifier for the atmospheric record.

3	Pressure	-	Unknown. Appears to always be empty.
3	Temperature	-	Unknown. Appears to always be empty.
3	PPM	-	Unknown. Appears to always be empty.
3	ApplyEarthCurvatureCorrection	-	Unknown. Appears to always be false.
3	ApplyRefractionCorrection	-	Unknown. Appears to always be false.
3	RefractionCoefficient	0	Unknown. Appears to always be 0.
3	PressureInputMethod	String	Unknown. Appears to always be ReadFromInstrument.

2	FlightMissionRecord {
	TimeStamp,
	ID
	}
		-	***Timestamp*** is the time the flight mission record was saved.
			
			ID is the internal identifier for the flight mission record.

3	Name	String	Name of the flight, which is used to create the JXL filename.

3	FlightBlock {
	endTime,
	name,
	startTime
	}
		-	

4	FlightPlan {
	percentLateralOverlap,
	percentForwardOverlap,
	height
	}
		-	

5	Node {
	latitude,
	longitude
	}
		-	

4	StationList	-	
5	StationID	Int	ID of each PhotoStationRecord for each image taken on this flight.
1	Reductions	-	Unknown. Appears to always be empty.

1	Environment	-	
2	DisplaySettings	-	Unknown. Appears to always be empty.
3	DistanceUnits	String	Unit of measure for distances. Appears to always be Metres.
3	HeightUnits	String	Unit of measure for heights. Appears to always be Metres.
3	AngleUnits	String	Unit for angles. Appears to always be DecimalDegrees.
3	AzimuthFormat	String	Unknown. Appears to always be Azimuth.
3	LatitudeLongitudeUnits	String	Unit of measure for lat/long. Appears to always be DecimalDegrees.
3	CoordinateOrder	String	Ordering for co-ordinates. Appears to always be North-East-Elevation.
3	TemperatureUnits	String	Unit of measure for temperature. Appears to always be Celsius.
3	PressureUnits	String	Unit of measure for pressure. Appears to always be MiliBar.
3	GradeUnits	String	Unknown. Appears to always be Percentage.
3	AreaUnits	String	Unit of measure for areas. Appears to always be SquareMetres.
3	StationFormat	String	Unknown. Appears to always be 1+000.0.
2	JobSettings	-	
3	NeighbourhoodAdjustment	-	Unknown.
4	Applied	Bool	Unknown. Appears to always be false.
4	WeightExponent	0	Unknown. Appears to always be 0.
2	TimeZone	-	Timezone information for the flight.
3	ZoneName	-	Unknown. Appears to always be empty.
3	HoursToUTC	Int	Difference in hours from UTC for the flight.
2	CoordinateSystem	-	
3	SystemName	String	Unknown. Appears to always be Default.
3	ZoneName	String	Unknown. Appears to always be Default.
3	DatumName	String	Unknown. Appears to always be WGS 1984.
3	Ellipsoid	-	
4	EarthRadius	Int	Radius of the Earth. Appears to always be 6378137.
4	Flattening	Float	Unknown. Appears to always be 0.00335281067183.

3	Projection	-	
4	Type	String	Unknown. Appears to always be NoProjection.
4	Scale	1	Unknown. Appears to always be 1.
4	GridOrientation	String	Unknown. Appears to always be IncreasingNorthEast.
4	SouthAzimuth	Bool	Unknown. Appears to always be false.
4	ApplySeaLevelCorrection	Bool	Unknown. Appears to always be false.
3	LocalSite	-	
4	Type	String	Unknown. Appears to always be Grid.
4	ProjectLocationLatitude	-	Unknown. Appears to always be empty.
4	ProjectLocationLongitude	-	Unknown. Appears to always be empty.
4	ProjectLocationHeight	-	Unknown. Appears to always be empty.
3	Datum	-	Information about the GPS location system used.
4	Type	String	Unknown. Appears to always be ThreeParameter.
4	GridName	String	Unknown. Appears to always be WGS 1984.
4	Direction	String	Unknown. Appears to to always be WGS84ToLocal.
4	TranslationX	0	Unknown. Appears to always be 0.
4	TranslationY	0	Unknown. Appears to always be 0.
4	TranslationZ	0	Unknown. Appears to always be 0.
3	HorizontalAdjustment	-	
4	Type	String	Unknown. Appears to always be NoAdjustment.
3	VerticalAdjustment	-	
4	Type	String	Unknown. Appears to always be NoAdjustment.
3	CombinedScaleFactor	-	
4	Location	-	
5	Latitude	-	Unknown. Appears to always be empty.
5	Longitude	-	Unknown. Appears to always be empty.
5	Height	-	Unknown. Appears to always be empty.
4	Scale	-	Unknown. Appears to always be empty.

Table 1: JXL Structure Definition

3.5. Non-Functional Requirements

This section describes the software requirements that are not related to the functionality of the software itself, but on the “usefulness” of the software for the environment it will be used.

3.5.1. Maintainability

The software shall be modifiable to allow for the potential to parse additional information from the JXL data structure. This means that the parser must be modular and the database must allow for additional elements, which may be null for older data without the additional information.

3.5.2. Portability

The software shall be used on a variety of operating systems, with Linux, Windows, and Mac OSX the primary targets. Therefore, the software shall be easily portable to allow for usage in variable environments.

3.6. Database Requirements

This section describes the database structure required by the software. The database shall be relational and shall require storage of both flight and image metadata. tblFlights has a 1-to-many relationship with tblImage, where a single flight has many images associated with the flight, while an image is only associated with a single flight.

3.6.1. Database structure

This is a high-level structure of the database. For the description of each element, see 3.6.2.

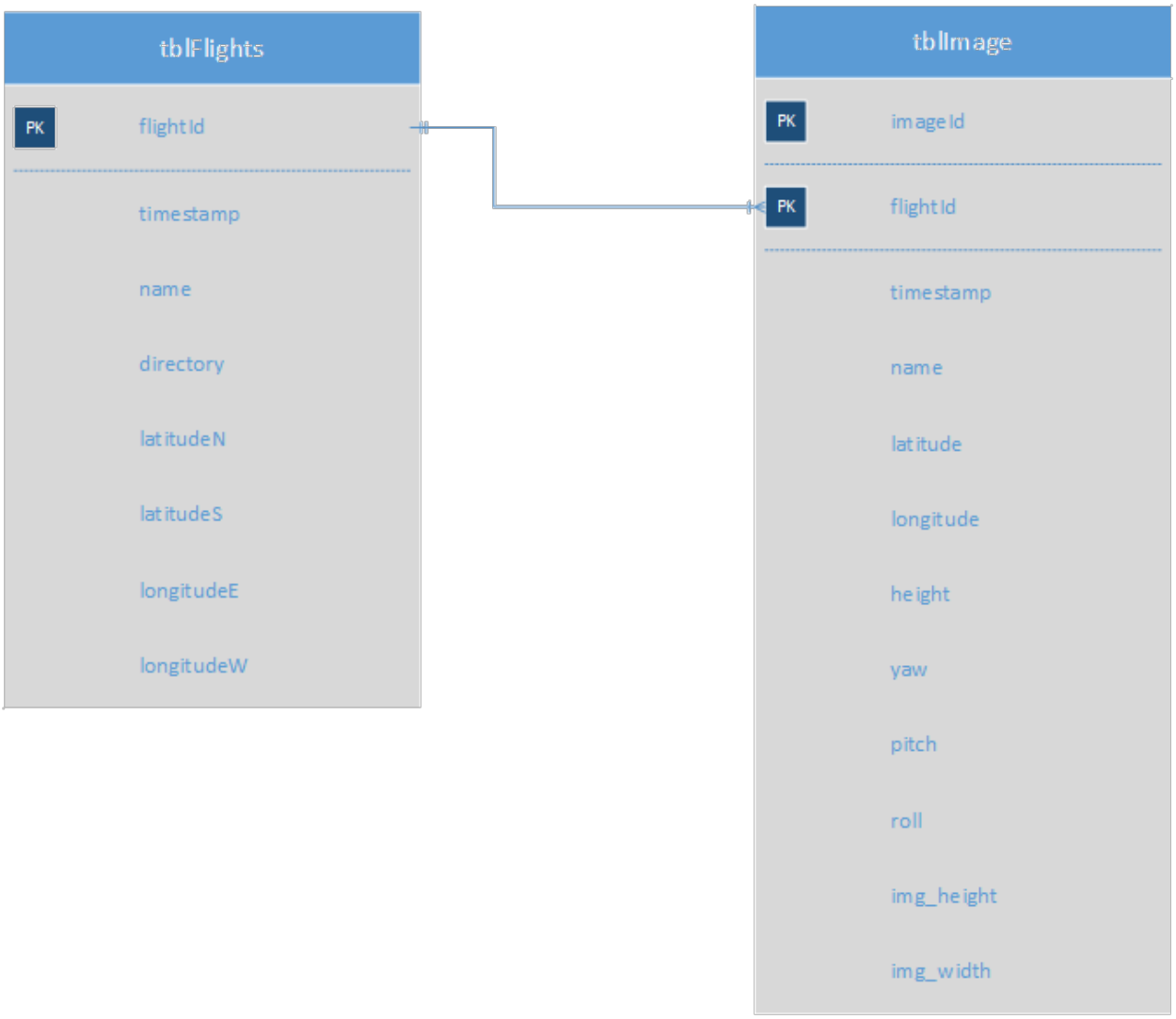

Figure 1: Database relational structure

3.6.2. Database description

This section describes each element for each table in the database.

3.6.3. tblFlights

tblFlights holds the information related to individual flights. No elements can be NULL.

Name	Type	NULL?	Description
flightId	PK int	N	Primary key
Timestamp	Datetime	N	Time of the flight
Name	Varchar	N	Name given to the flight
Directory	Varchar	N	Directory where the images are stored
latitudeN	Float	N	Most northern latitude of the flight
latitudeS	Float	N	Most southern latitude of the flight
longitudeE	Float	N	Most eastern longitude of the flight
longitudeW	Float	N	Most western longitude of the flight

Table 2: tblFlights description

3.6.4. tblImages

tblImages holds the information related to the images from a flight. No elements can be NULL.

Name	Type	NULL?	Description
imageId	PK int	N	Primary key
flightId	FK int	N	Foreign key to tblFlights
Timestamp	Datetime	N	Time of the image
Name	Varchar	N	Name given to the image
Latitude	Float	N	Latitude of the image
Latitude	Float	N	Longitude of the image
Height	Float	N	Height from which the image was taken
yaw	Float	N	Yaw when the image was taken
Pitch	Float	N	Pitch when the image was taken
Roll	Float	N	Roll when the image was taken
Img_height	Int	N	Height of the stored image
Img_width	Int	N	Width of the stored image

Table 3: tblImages description