Abstract. A new software tool for the detection and monitoring of plumes caused by major technological accidents is described. The objectives of this tool are to use real time information as provided by satellite images, to use sophisticated image processing techniques and to achieve a user friendly operational environment for the detection of plumes caused by major technological accidents. The methodology for the automatic detection and monitoring of plumes caused by major technological accidents on NOAA/AVHRR (Advanced Very High Resolution Radiometer) imagery, uses a two-dimensional feature space in order to discriminate pixels that contain plumes from those correspond to clouds or to the underlying surface. The two-dimensional feature space is generated by combining AVHRR channels 1, 2 and 5. The proposed software tool has been coded in JAVA2 language using the concepts of interoperability and object-oriented programming. This study demonstrates the applicability of the proposed tool for the detection of a plume caused by a massive explosion in a firework factory in Enschede, The Netherlands, on May 13, 2000.

1. Introduction
Technological accidents can be characterized by a number of different events and processes, including spillage or sudden release of materials, fire, or explosion. The most common effect is the release of gases and liquids used and processed in the installations concerned. Fire and explosion are also common effects, while a combination of the above is not rare. Releases may be toxic and can be either to land and water or to air. Airborne releases usually develop in plumes, which can thereafter be monitored either due to their optical depth or their temperature difference from the ambient air. Damages may thus occur both as an immediate and direct consequence of the accident, and subsequently during propagation and dispersion of the resulting plume.
It should be mentioned that while exhaustive consideration has been given to the immediate ground-level effects in close vicinity to the installation, only limited effort is usually given to examining the impacts of the plume in the wider geographic area during the course of the hours or days following the accident. In recent years, and due to a number of incidents involving fires in industrial installations and warehouses, research has been oriented towards the definition of the properties and of the amount of the plume particulates generated by different materials, including pesticides, under varying fire conditions (Lang 1993, Bartelds et al. 1993, Atkinson and Jagger 1994, Miles and Cox 1994, Grant and Drysdale 1994, Martins and Borrego 1994, Marliere, 1996, Porter et al., 1996, Martins et al. 1996, Cozzani et al. 1996).

In several studies, (Kaufman et al. 1990, Ferrare et al. 1990, Kaufman et al. 1992, Cahoon et al. 1994) satellite data have been used for the analysis of the emitted smoke in order to quantify the gaseous output from forest fires. Ackerman and Toon (1981), Kaufman (1987) and Fraser et al. (1984), related the carbon content of the plume to the single scattering albedo, as well as to the light extinction of the plume. Chung and Lee (1984) examined the feasibility of using satellite imagery to detect large-scale pollution episodes. Christopher and Chou (1997) used a combination of spectral and textural measures in order to separate visually the plume aerosols from the underline background. Baum and Trepre (1999) proposed a grouped threshold method for scene identification in NOAA/AVHRR imagery that may contain clouds, fires, smoke plumes or snow. Chrysoulakis and Cartalis (2003a) proposed a software tool for the detection of major fires caused by technological accidents with the use of NOAA/AVHRR (Advanced Very High Resolution Radiometer) imagery.

In this paper a new software tool for the detection and monitoring of plumes caused by major technological accidents is described. The software is named JPLUME and is coded in JAVA2 language using the JDK (Java Development Toolkit) 1.2.2 (Sun 1998). JPLUME has been evaluated for four technological accidents: in Enschede, the Netherlands on May 13, 2000; in Genoa, Italy on April 13, 1991; in Lyon, France on June 2, 1987 and in Kalohori, Greece on February 24, 1986. In this study, an AVHRR image acquired over the broader area of Netherlands on May 13, 2000 (14.44 UTC) is used to present the functionality of the proposed software tool. This date refers to the massive explosion in a firework factory in the town of Enschede, The Netherlands.

In the past, software packages have been developed for technological risk assessment; some of these software packages are WHZAN (Technica 1992), RISKIT (VVT 1993), EFFECTS (TNO 1991), SAVE (TNO 1992) and MAXCRED (Khan and Abbasi 1999). JPLUME differs from the existing software packages in the following areas:

(a) it is a detection rather than a forecasting tool;
(b) it is not limited to the specific location of an industrial complex, but it is scalable; to this end it may be applied for the monitoring of areas covering a single industrial installation, as well as for extended areas;
(c) as a JAVA developed program it is independent of the hardware and the platform, a fact which provides interoperability to the software; additionally, users can easily modify the source code of the program using the JDK 1.2.2 which is freeware (Sun 1998);
JPLUME offers a window based user interface and is user friendly. It carries menus, buttons and a sketch of the software's algorithm as well as an Image Viewer which gives the opportunity to the user to track the various steps of the algorithm.

2. The JPLUME Methodology
The inputs of JPLUME are NOAA/AVHRR images. AVHRR has a spatial resolution of 1.1 km at the nadir, a temporal resolution of approximately six hours (for both the ascending and descending NOAA nodes) and a swath coverage of 2700 km. AVHRR records incoming radiation in five spectral channels (µm): 0.58-0.68 (visible), 0.72-1.10 (near infrared), 3.55-3.93 (mid-infrared), 10.5-11.3 (thermal infrared) and 11.5-12.5 (thermal infrared).

An explanation of the physical background of the procedure used to differentiate between plumes and clouds, as well as between plumes and the underlying surface has been given in previous studies (Chrysoulakis 2000, Chrysoulakis and Cartalis 2003b). The algorithm used in JPLUME carries the advantages of multispectral analysis and provides valuable results for the detection and monitoring of plumes caused by technological accidents. The main stages of the algorithm are:

(a) Calibration of channels 1, 2 and 5 of the AVHRR image. The calibration procedure is based on the conversion of the digital numbers to brightness temperatures for infrared channel (5) and to reflectance values for the visible channels (1 and 2), on the basis of lookup tables. Values of lookup tables have been calculated using the equations given in the NOAA Polar Orbiter Data Users Guide (Kidwell 1997).

(b) Production of a pseudochannel image of a cloud masking filter named CLD (Chrysoulakis and Cartalis 2003b). CLD is used for the separation of pixels that contain plume from the pixels correspond to clouds.

(c) Production of a pseudochannel image of the well known NDVI (Normilized Difference Vegetation Index). NDVI is used for the separation of pixels that contain plume from the pixels correspond to the underneath surface.

(d) Application of the CLD and NDVI digital filters and production of a pseudochannel image of the two dimensional feature space which is produced with the use of both filtered pseudochannels CLD and NDVI.

(e) Search for pixels with values greater than a given threshold in the pseudochannel image of the CLD and for pixels with values lower than a given threshold in the pseudochannel image of the NDVI. Pixels which meet both the above mentioned conditions are classified as plumes (Chrysoulakis and Cartalis 2000b).

Each of the above mentioned stages of the algorithm is implemented by a different module of JPLUME. In particular each module consists of one or more programme classes, which have been programmed using the object oriented JAVA2 language. The fundamental unit in object-oriented programming is the "object". Languages which follow object-oriented concepts describe the interaction among objects. There are no stand-alone constants, variables or functions; everything is accessed through classes and objects. A class is a type definition, whereas an object is a variable declaration. Classes encapsulates objects, whereas a single class can be used to instantiate multiple objects. The image manipulation in JPLUME is implemented with the use of the immediate mode imaging model of JAVA 2DTM API. The JAVA 2DTM API is a set of...
classes for advanced two-dimensional graphics and imaging. The immediate mode imaging model supports fixed-resolution images stored in memory. The model also supports filtering operations on image data. A number of classes and interfaces are used in this model (Sun 1998). JPLUME converts the original AVHRR channels to “BufferedImage” objects with the use of the capabilities of BufferedImage class. A BufferedImage object contains two other objects: a "Raster" and a "ColorModel". The Raster class provides image data management. It represents the rectangular coordinates of the image, maintains image data in memory, and provides a mechanism for creating multiple sub-images from a single image data buffer. It also provides methods for accessing specific pixels within an image. A Raster object contains two other objects: a "SampleModel" and a "DataBuffer". The SampleModel class interprets data in the buffer and provides it as individual pixels or rectangular ranges of pixels. The DataBuffer class holds pixel data in memory and the access on these data is implemented in byte level. Taking into account that AVHRR images are structured in 10 bits format (2^{10} = 1024 grey levels for each pixel), it is obvious that 16 bits (2 bytes) are used for the representation of each pixel. Therefore, each AVHRR channel is stored in memory as a 16 bit image which contains unsigned short values. One dimensional arrays are used for this storage. The digital number of the i element of a DataBuffer object (DN(i)) corresponds to the digital number of the pixel with coordinates (x, y) of the respective Raster object. The correspondence is defined using the formula:

\[ DN_{(i)} = DN_{ym_{W+x}} \]

where W is the number of columns of the AVHRR image.

The ColorModel class provides a colour interpretation of pixel data provided by the sample model of the image. It defines methods for converting a pixel value to a colour value in its associated colour space. The class "ColorSpace" provides predefined ColorSpace objects for any type of image. The dimensions of the colour space are defined by the number of basic colours used as components (Pantham 2000).

3. Design and Application of JPLUME

In the following section JPLUME is described in detail with regard to its design and application. Figure 1 represents the main window which presents the available options in JPLUME. A sketch of the main stages of the algorithm is presented in the right part of this window. The sketch informs the user about the steps of the algorithm. The series of buttons presented in the left part of the main window activates the modules which implement the various parts of the JPLUME algorithm. Each module includes one or more classes of the object-oriented code. Each class creates one or more objects which are called by the programme on the appropriate time.

Figure 2 shows the object-oriented architecture of JPLUME which is organised in four essential stages. An image viewer is also included in JPLUME which is activated in a separate window. Input and output images as well as intermediate image products can be presented with the use of this viewer. For the application of JPLUME, three image input files (AVHRR channels 1, 2 and 5) and three input calibration files are necessary. Image files are raw NOAA/AVHRR data which can be retrieved from NOAA ground receiving stations. They also can be extracted from the standard NOAA Level 1b format by removing all metadata information. All image files must have the suffix .nek in order to be recognised by JPLUME (i.e. channel5.nek). Each file in .nek
format contains a single AVHRR channel which was provided from the geometric corrected AVHRR original image. All files must be uncompressed with no metadata information, whereas pixels in .nek format have been coded in two bytes words (16 bit images). The AVHRR image of May 13, 2000 (14.44 UTC) was used in this paper for the application of JPLUME.

Figure 1. The main window of JPLUME.

The remaining three input files are the calibration files which are provided in ASCII format and obtain the suffix .txt (text files). Each .txt file contains 1024 rows. The number of each row corresponds to a grey level of the 10 bits images (0 - 1023). Each row consists of the brightness temperature or reflectance values which have been calculated using the equations given in the NOAA Polar Orbiter Data Users Guide (Kidwell 1997).

On the basis of Figures 1 and 2, the functionality of each button is described by analysing the function and the structure of the module of JPLUME, as activated by pressing the respective button.
Figure 2. The object-oriented architecture of JPLUME: The essential modules of the programme as well as their inputs and outputs are presented. C1, C2 and C5 represent the AVHRR channels 1, 2 and 5, respectively; T1, T2 and T5 represent the calibration files for channels 1, 2, 5, respectively; CC1, CC2 and CC5 represent the calibrated channels 1, 2 and 5, respectively; CLD* represents the pseudochannel image of the normalized ratio of channels 1 and 5 filtered by itself; NDVI* represents the pseudochannel image of the normalized ratio of channels 1 and 2 filtered by itself; CLD represents the CLD* image filtered by NDVI*; NDVI represents the NDVI* image filtered by CLD* and FS represents the two dimensional feature space image which is produced from CLD and NDVI pseudochannels.
3.1. **Image Viewer Module**
This module is activated by pressing the button "Image Viewer". Following, a new window is launched containing buttons which allow the projection of input or intermediate image files. Ten different classes have been coded for the presentation of these images on screen. Figure 3 shows how the AVHRR channel 5 image of May 13, 2000 (14.44 UTC) is presented in this Viewer.

![AVHRR Channel 5 Image](image.png)

**Figure 3.** Presentation of the AVHRR channel 5 image of May 13, 2000 (14.44 UTC) with the use of Image Viewer module of JPLUME

3.2. **Calibration Module**
This module is activated by pressing the button "Calibration". Following, a new window is launched containing buttons which allow the calibration of each input image file. Three different classes have been coded for the image calibration. The calibration procedure for the AVHRR channel 5 is presented graphically in Figure 4. The digital number of each pixel of the original image is used as index in the lookup table included into the respective calibration file in order to specify the respective brightness temperature value. This value is assigned to a pixel of a new Buffered Image. The position (row, column) of this pixel in the new image is the same with the position of the original pixel in the AVHRR image. Therefore, during the calibration procedure, a new Buffered Image is created for each AVHRR channel. For example, the calibration procedure for channel 5 receives as inputs the files channel5.nek and table5.txt and generates the file channel5calibr.nek.

![AVHRR Channel 5 Calibration](calibration_image.png)
3.3. Step 1 Module

This module includes two sub-modules: The “CLD (Step 1)” and the “NDVI (Step 1)”. CLD (Step 1) is activated by pressing the respective button. The JPLUME class which is called, is executed at the background using the calibrated channels 1 and 5 files in order to produce CLD*. The produced file receives the name cld1.nek. For each pixel in this pseudochannel image, CLD* value is given by the formula:

\[
CLD^* = \frac{BT_5 - R_1}{BT_5 + R_1}
\]  

where:
- BT\(_5\) is the AVHRR channel 5 brightness temperature;
- R\(_1\) is the AVHRR channel 1 reflectance.

Chrysoulakis and Cartalis (2000, 2003b) have shown that CLD* values lower than a given threshold correspond to clouds. CLD* thresholds from 0.85 to 0.95 can be used for the cloud filtering in JPLUME. In practice, pixel values lower than 0.90 are setting to zero by CLD (Step 1) sub-module. Figure 5 shows the final CLD* image.

NDVI (Step 1) is activated by pressing the respective button. The JPLUME class which is called, is executed at the background using the calibrated channels 1 and 2 files in order to produce NDVI*. The produced file receives the name ndvi1.nek. For each pixel in this pseudochannel image, NDVI* value is given by the formula:

\[
NDVI^* = \frac{R_2 - R_1}{R_2 + R_1}
\]

where R\(_2\) and R\(_1\) are the AVHRR channel 2 and 1 reflectances, respectively.

Negative NDVI* values threshold correspond to water bodies or clouds (Chrysoulakis and Cartalis 2003b). In JPLUME, negative pixel values setting to zero by NDVI (Step 1) sub-module.
3.4. **Step 2 Module**

This module produces CLD and NDVI pseudochannels filtered both for clouds and water bodies. It includes two sub-modules: The “CLD (Step 2)” and the “NDVI (Step 2)”. CLD (Step 2) is activated by pressing the respective button. The JPLUME class which is called, is executed at the background uses files cld1.nek and ndvi1.nek in order to filter CLD* pseudochannel for water bodies and clouds. Pixels with zero NDVI* values are used to produce a digital mask applicable in this filtering process. The produced file receives the name cld2.nek. Figure 6 shows the final CLD image.

![Production of AVHRR Channels 1 and 6 CLD = (Ch5 - Ch1)/(Ch6 + Ch1) Step1](image1)

**Figure 5.** CLD* pseudochannel image, a product of Step 1 Module.

![Production of AVHRR Channels 1 and 6 CLD = (Ch5 - Ch1)/(Ch6 + Ch1) Step1](image2)

**Figure 6.** CLD pseudochannel image, a product of Step 2 Module.
NDVI (Step 2) is activated by pressing the respective button. The JPLUME class which is called, is executed at the background uses files cld1.nek and ndvi1.nek in order to filter NDVI* pseudochannel for clouds. Pixels with zero CLD* values are used to produce a digital mask applicable in this filtering process. The produced file receives the name ndvi2.nek.

3.5. **Plume Module**

![Image of Plume Module](https://via.placeholder.com/150)

**Figure 7.** Pseudo-coloured image RGB: CLD, NDVI, 0, which is the result of the Plume Module of JPLUME. This pseudo-coloured composition represents the two-dimensional feature space CLD – NDVI. The pixels in heavy red in the vicinity of Enschede (within the white circle) correspond to the plume caused by the massive explosion in the firework factory.

This module is activated by pressing the button "Plume Detection". Following, a new window is launched containing two different buttons. Each button activates one sub-module, respectively. The two sub-modules included in this module are: The “Plume Detection and Monitoring” and the “Export”. Plume Detection and Monitoring uses cld2.nek and ndvi2.nek files and allows the production of the final JPLUME product and its projection on the screen. The final JPLUME product is a two dimensional feature space (CLD – NDVI) image on which the plumes can be detected and monitored. The two dimensional feature space is developed by superimposing the pseudochannels CLD and NDVI. Practically, this superimposition is achieved by creating a pseudo-coloured image RGB: CLD, NDVI, 0; in particular the double-masked CLD pseudochannel was set to RED, the double-masked NDVI pseudochannel was set to GREEN, whereas all values of the BLUE channel were set to zero. Chrysoulakis and Cartalis (2003b) have shown that the CLD values of pixels that contain plumes are high, whereas the NDVI values are relatively low. Consequently, plumes should appear in heavy red colour in the pseudo-coloured image. Similarly, the CLD values of pixels that contain land surfaces are high as well as their
NDVI values. Therefore, land surfaces should appear in green colour in the pseudo-coloured image. In Figure 7 this pseudo-coloured image is presented, as resulting from the application of JPLUME to the AVHRR image of May 13, 2000. The pixels in red in the vicinity of Enschede area (within the white circle) correspond to the plume caused by the massive explosion in the fireworks factory. These pixels can be easily separated in the CLD - NDVI space, therefore the plume can be detected and monitored.

The final pseudo-coloured image can be saved on the hard disk with the use of the Export sub-module. The Band Sequential technique is used for the storing of the three bands (CLD, NDVI, 0) of the pseudo-coloured image in a single file which receives the name plume.nek. This file can be opened in any image processing software as generic binary file by giving the following three parameters: a) the number of rows (H); b) the number of columns (W); c) the number of bands (3).

5. Conclusions

The detection of a plume caused by a major technological accident with the use of satellite imagery relates to its dispersion, optical thickness and temperature structure, as well as to the spatial resolution of the satellite sensor. A new satellite images analysis software tool, the JPLUME, has been developed as a comprehensive and user-friendly tool for the automatic detection and monitoring of plumes caused by major technological accidents. JPLUME methodology is based on the development of two-dimensional feature space image in order to discriminate pixels that contain plumes from those correspond to clouds or to the underlying surface. This two-dimensional feature space is generated by combining the normalised ratios 

\[(5-1)/(5+1) \] and \[(2-1)/(2+1)\] of AVHRR channels 1, 2 and 5. The first normalised ratio (named CLD) takes into account the received radiation in the visible and thermal infrared. It is used for the discrimination of pixels that contain plumes from those that contain clouds and for the masking of cloudy pixels. The second normalised ratio (the well known NDVI) takes into account the received radiation in the visible and near infrared. It is used for the discrimination of pixels that contain plumes from those corresponding to the underlying surface, as well as for the masking of pixels that contain water bodies. JPLUME, coded in JAVA2 programming language, has the following attributes:

(a) effectiveness and automatic operation: JPLUME can be used for the real time detection and monitoring of plumes in real situations because its methodology is exclusively based on the use of satellite imagery;

(a) reliability: JPLUME's reliability is dependent on the spatial and temporal resolution of the satellite data used as inputs;

(b) scalability: JPLUME can be applied for the monitoring of an area covering a single industrial installation, as well as for the monitoring of areas covering half of Europe, depending on the available satellite data;

(c) interoperability: JPLUME, as a JAVA developed application, is independent of the platform and the operating system;

(d) user-friendliness: JPLUME has a window based user interface. It carries menus, buttons, a sketch of the application algorithm and an Image Viewer on which the input and output images as well as the intermediate products can be presented.
The functionality and the applicability of JPLUME were demonstrated with an illustrative example of a technological accident in a firework factory in the town of Enschede (The Netherlands) on May 13, 2000. The AVHRR image of May 13, 2000 (14.44 UTC) has been used in JPLUME. The effectiveness and reliability of JPLUME may be improved if it is combined with other sources of information, such as information regarding the fire caused by a technological accident. Such information may be retrieved by the application of fire detection algorithms to AVHRR imagery. Finally, it should be mentioned that high and very high spatial resolution satellite imagery can be used for the site location and mapping of major industrial installations for each area of concern.

References


