TABLE OF CONTENTS

REPORT

Building a Scale Score
Submitted By:
CENTRE FOR BUILDING PERFORMANCE RESEARCH
To:
Opus International Consultants

Introduction

To arrive at a scale score for an Urban Environment this project combines sun, wind and temperature data in an index of the liveability for public spaces. This index can be used and applied to a proposed change for an inner urban area. The index will take into account the comfort of users of the space and the priorities they have for that environment.

This brief report records the process by which a scale score is arrived at, including:

? Raw data used;
? computer programs used and their limitations;
? data transformations;
? problems and limitations encountered at each stage of the Scale Score method.

It must be made clear that this report outlines a test that has been performed on data from Midland Park, and may not apply to other urban areas until further testing has been done.

The report is divided into five main sections. The first section (Method Flow Chart) provides a quick guide to the process that is used to arrive at a scale score.

Section 2.0 (CAD Model), looks at how the Computer Aided Design Model was constructed and what this model represents. This section describes in detail how to calculate a sun/shade ratio for any given space, and the difficulties and concerns in this process.

The third section (Weather Data) consists of the step by step process for using a National Institute for Water and Atmospheric Research (NIWA) weather file to predict scale score values for an urban space. This section also includes any issues that have arisen that can influence the outcome of this process.

Section 4.0 (Urban Score), combines the information from the previous two sections to arrive at a scale score.

Lastly, the conclusions section sums up the whole process and suggests whether or not the method used does in fact come up with a valid scale score for Midland Park.

1.0 Method Flow Chart

The following flow chart (Figure 1) shows all the inputs that go into the scale score calculations. Each step in the process is described in detail in the following sections of this report.

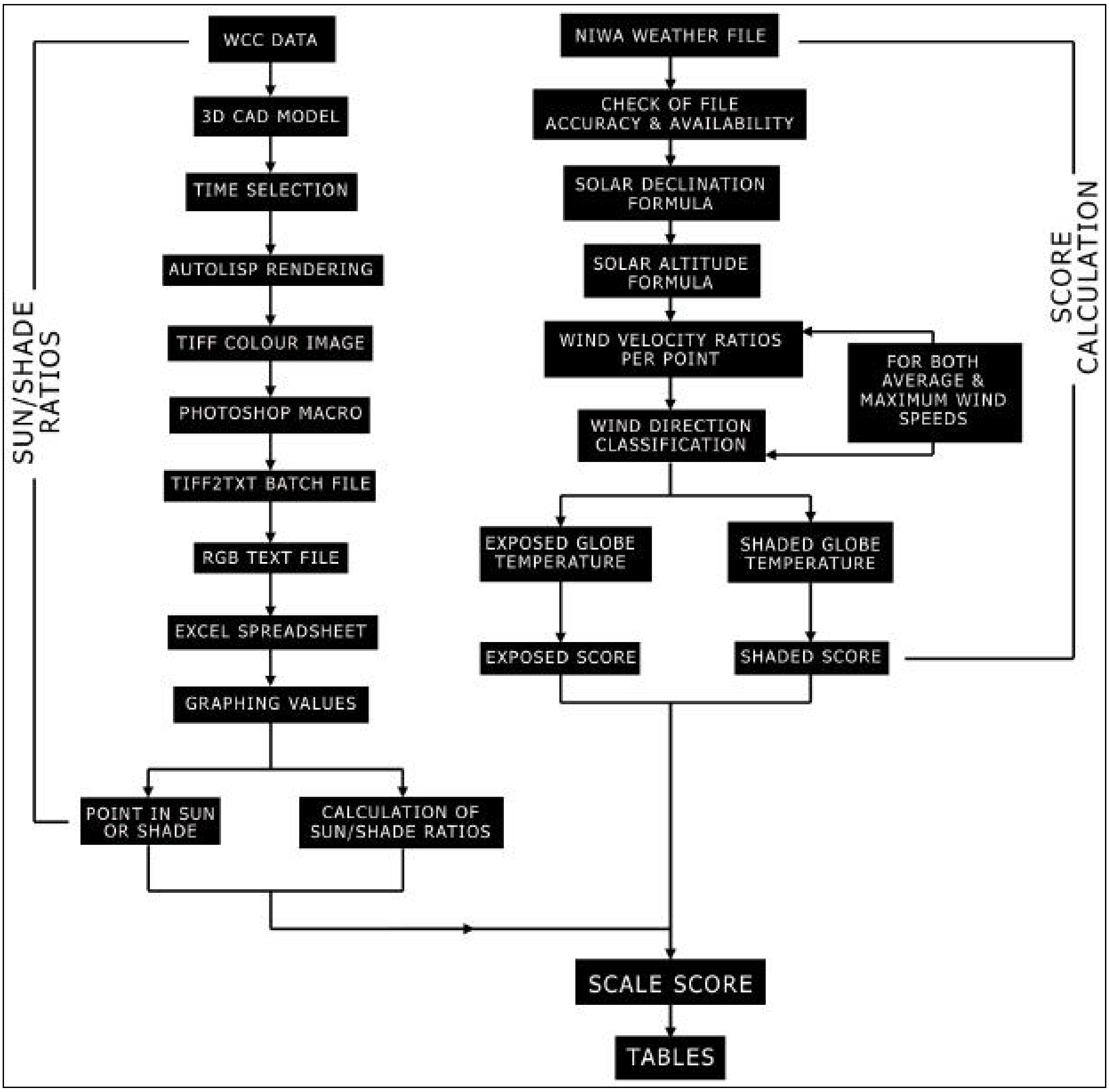


Figure 1 - Method Flow Chart

2.0 CAD Model

A three dimensional Computer Aided Design (CAD) model of Wellington City was used to analyse solar data. Figure 2 shows the model for Midland Park both in plan and perspective views.

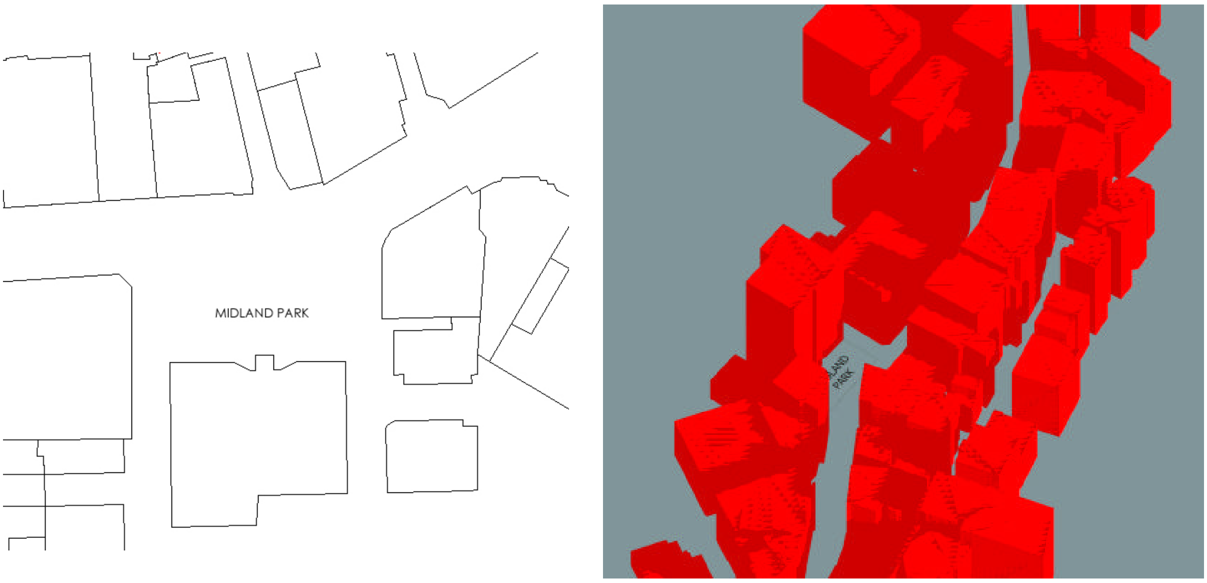


Figure 2 - Midland Park CAD Model

CAD files were obtained from:
Mike Reed, Asset Information, Wellington City Council
Email: **Reed1M@wcc.govt.nz**
Phone: (04) 801-3637

The CAD files are accurate as of the late 1990's and consist of the building stock of Wellington City. The information included in the files is:

? Buildings – plans with building heights appended
? Kerb and Seal
? Fence
? 1m Contour Lines
? Aerial Photos (1:500 & 1:5000)

All data is provided on CD. Vector data is provided in dxf format and/or shape files. Image data is provided in tiff or jpeg format. A world file is also provided; this contains the New Zealand Mapping Grid (NZMG) coordinates of the bottom left hand corner of the image so it is easily geo-referenced.

Figure 2, shows the model of Midland Park, a 3D model of most of Wellington's CBD has been constructed (Lambton Quay through to Te Aro). Figure 3 (next page) shows a representation of areas constructed in CAD.

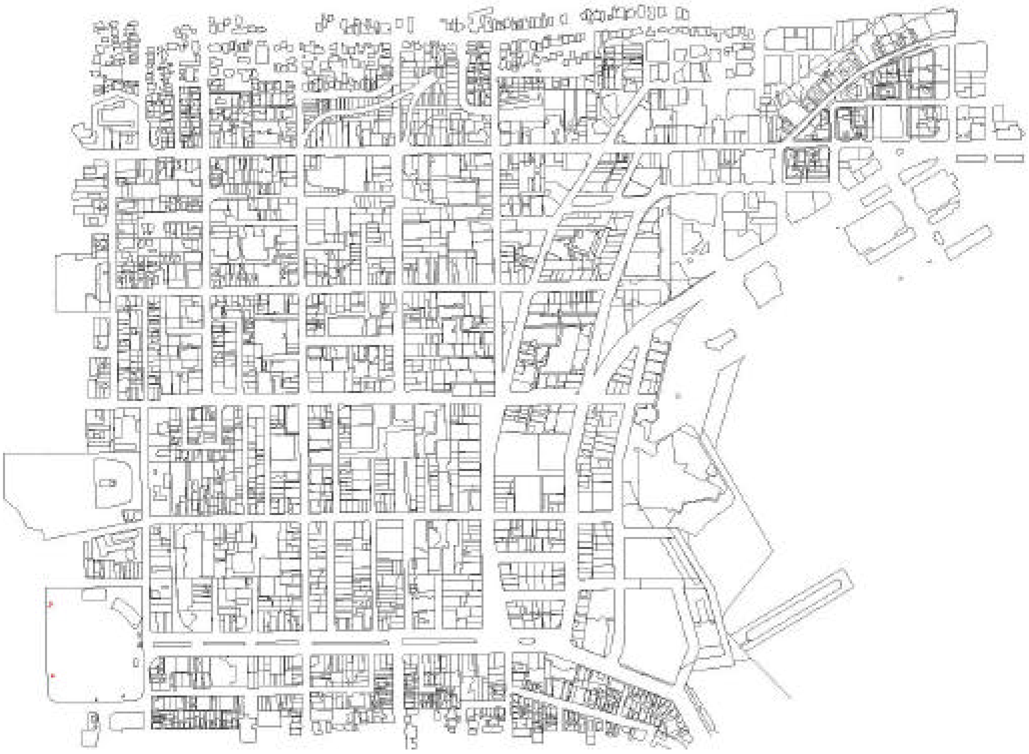


Figure 3 - CAD Model of Wellington

2.1 Time Selection

Before any simulations are made, the time of day and year need to be selected. There is no limit to the number of simulations made. For means of testing the Midland Park test case; January 21, April 21, July 21 and October 21 were the four dates selected as being representative of the whole year.

The times of the day that were considered for solar access were 1000hrs, 1200hrs and 1500hrs. These times were selected as they represent a simple cross-section of the day.

2.2 AutoLISP Rendering

In the CAD program, a single "distant" light simulates the sun. Although the sun radiates in all directions, because of its size and distance, by the time its rays reach the earth they are effectively parallel. Because a distant light is so frequently used to simulate the sun in this way, especially in architectural renderings, the photorealistic renderers provide a special sun angle calculator that calculates the sun's position based on both the hour of the day and geographic location.[1]

An AutoLISP routine 'SOLRENDER' was created to automate the solar access 'tests'. The user inputs the dates and the time interval required and the routine instructs the computer to carry out the renderings and produce images for each time of the day and time of the year. This makes it feasible to produce images for many more times than is conventionally possible with hand held calculators.

[1] AutoCAD Help: User Documentation

Increasingly this type of automation is being provided as an in-built tool within the CAD program. See Appendix 1 for SOLRENDER routine.

The SOLRENDER routine renders out colour Tiff images to a specified directory. Before any analyses can be performed on the data from these renderings, the image must be transformed into a format that Microsoft Excel or SPSS can understand.

2.2.1 Photoshop Macro

For later conversion into a Text file, the image needs to be in grayscale tiff format (not indexed). To do this a macro has been created in Photoshop that automates this process and turns the image(s) created by SOLRENDER into un-indexed, greyscale tiff images.

Now the Tiff2Txt batch conversion process can be used.

2.3 Tiff to Text File Conversion

Once the files have been rendered a batch file has been created that uses two programs from the Rayfront platform; ra_tiff.exe and pvalue.exe. See Appendix 2 for explanations of ra_tiff.exe and pvalue.exe. The batch file converts the tiff images created by SOLRENDER routine into a text file that can be imported into Excel or SPSS. The text file contains an ASCII data array showing x, y, r, g and b values, the output is 5 columns of numbers (X, Y, R, G and B) ahs values.

This conversion then allows the percentage area in shade and the percentage area in sun to be calculated by the RGB value of sun and shade colours using Microsoft Excel or SPSS.

2.4 Excel Spreadsheet – Pixel Counting

Excel was used to calculate the percentage area in sun and percentage area in shade from the text files created earlier. This was done by determining the RGB value for each type of area; shade, sun and building from the text file. A graph was drawn using the Midland Park example illustrating the values and suitable cut-off points were made for each area. An example graph can be seen in Figure 4.

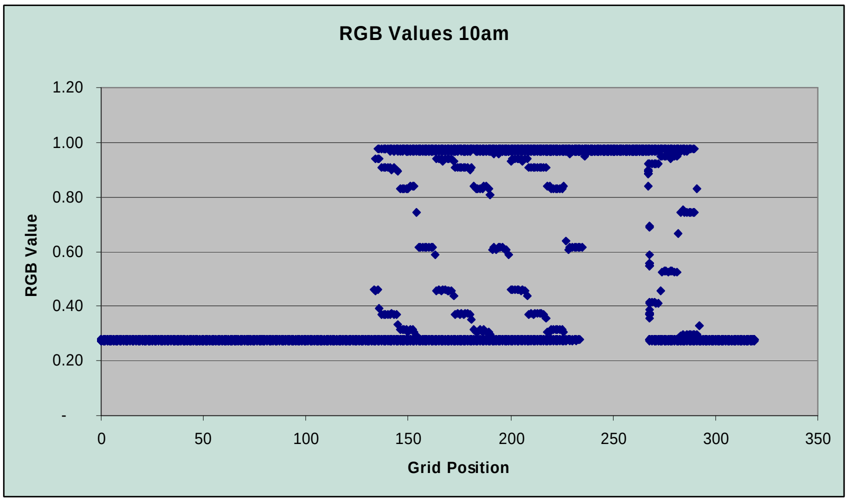


Figure 4 - RGB Graph

Cut-off RGB values were:		
	Building	< 0.27
	Shade	0.27 - 0.97
	Sun	> 0.97

This process was completed for each time of day (1000, 1230 and 1500 hours) on the 22 December (Summer Solstice). These tabulations resulted in tables comprising of the "Sun" and "Shade" percentages for each time of the day. The summary of these values for Midland Park can be seen below in Table 1.

Table 1 - Summary Table of Sun & Shade Values for Midland Park

	1000 hours Percentage	1230 hours Percentage	1500 hours Percentage
Sun	17%	71%	63%
Shade	69%	16%	24%

3.0 Weather Data

The first step in calculating a score is to obtain a NIWA data file. This file comprised of some 30 years of data, equalling to over 250,000 lines of data to be dealt with. The weather file contained the following variables; year, month, day, day number, temperature, solar radiation, maximum airport wind, average airport wind, and airport wind direction.

The scale score equation requires the following variables: average wind, maximum wind, ambient temperature, and globe thermometer temperature. Ambient temperature is able to be used directly from the raw weather file; however the two wind parameters and the globe thermometer must be calculated through a series of steps.

3.1 Test of Weather File Accuracy

Data for Solar Radiation, Temperature, Wind Speeds and Directions recorded at Wellington Airport and Kelburn over the past 30 years was obtained from NIWA's National Climate Database. The National Climate Database (CliDB) is New Zealand's national repository of high quality climate data. It comprises historical paper-based archives dating back to 1860. The program operates under ISO 9002 accreditation and includes "development efforts to respond to ever-changing user demands and the opportunities offered by new technology".[2]

Investigation was done into the availability of data from the three New Zealand locations concerned (Auckland, Wellington and Christchurch), as some of the weather stations in New Zealand were either out of commission or had only being recording for a few years. The project called for 20-30 years of data, so that the averages concluded were a good indication of happenings in that area.

The data from CliDB comes in text format and was changed into a format readable by SPSS. SPSS provides a powerful set of sophisticated univariate and multivariate analytical techniques for real-world problems. It was used for its ability to deal with large volumes of information (the Wellington file comprises some 250,000 lines of data). By following a system of taking the raw data from CliDB into a text editing program (Kedit), the data was separated into columns which could then be imported directly into SPSS with little further formatting required.

The data obtained at Auckland, Wellington and Christchurch is from the weather stations at their respective airports, and the frequency of the measurements was hourly. Early checks were made to ensure the accuracy of the data, and a cross tabulation was done comparing year/month, year/day and year/hour. These cross tabulations made it clear that data was missing. However, as less than 10% of the data was missing, it was determined to have a negligible effect on the outcome of the tests.

[2] http://www.niwa.co.nz/services/clidb/ Accessed 29 August 2002

3.2 Wind Tunnel Test Data

Existing wind tunnel test reports were used as sources of information for the purposes of constructing the index. The wind tunnel studies have been completed by Opus Central Laboratories.

These reports consist of flow visualization tests that use erosion of polystyrene beads to establish a relative degree of windiness in a proposed development area. Hot-film anemometers were used to obtain wind speeds at various locations around the site, particularly in areas where there are foreseeable problems.

The information used from within the reports completed by Opus Central Laboratories was the Bead Speed Tests and wind speed point measurements. The bead clearance is indicated by varying colours relating to the speed at which the beads cleared in that particular area, thus showing the areas most prone to uncomfortable wind disturbance. The point measurements which were gained from the hot-film anemometers are displayed with a number corresponding to the wind speed in that area.

Locke tested the accuracy of bead speed tests, which have been used from the wind reports (See Appendix 4). Locke drew contour maps in much the same way as above and found that the least accurate part of the analysis was assigning 'frequencies' to the locations where wind speeds were measured with the hot-film probe. Locke found that two problems arose here:

1. Identifying the location on the image.

2. Interpolating a frequency value from the contours.[3]

3.2.1 Wind Ratios & Classification

The NIWA weather file contains a record of Wellington Airport wind speeds. Wind is severely manipulated by the urban environment, meaning that raw airport wind data can not be used to calculate a scale score in urban environments.

In order to predict the local site wind conditions from NIWA wind data, wind velocity ratios need to be applied to the airport wind speeds. The ratios express the wind speeds at a particular point in an urban space as a percentage of the airport wind speed. For each point, there are different ratios for each of the four main wind directions (170°, 200°, 320° and 350°).

The number of calculations that need to be done depends on the number of points being analysed. There are normally five calculations for each point, for both maximum and average wind. There is a calculation for each wind direction, each with an "if" statement. This results in four columns of data, so a further calculation is needed to condense them into one column.

Previous wind tunnel tests conducted only measured four wind directions, however in reality wind blows from all directions. OPUS is now measuring eight different wind directions, thus increasing the accuracy of these predictions. However, the wind tunnel test report used for the test outlined in this report consisted of only four wind directions. This means a decision must be made, as to how much of the compass each ratio is represent. An accurate means of division is to use small ranges, for example only apply ratios to wind which comes from a direction plus of minus ten degrees of the

[3] *Erosion Contour and Hotfilm Wind Speeds*, Nick Locke

“ratio direction” (170º, 200º, 320º or 350º). For example the 170º wind ratio is only applied to wind coming from between 160º and 180º.

Considering only winds within these small ranges reduces the number of cases that can be used for the analysis, which may result in a biased sample. The following (Figure 5) pie chart shows how hourly wind directions are distributed.

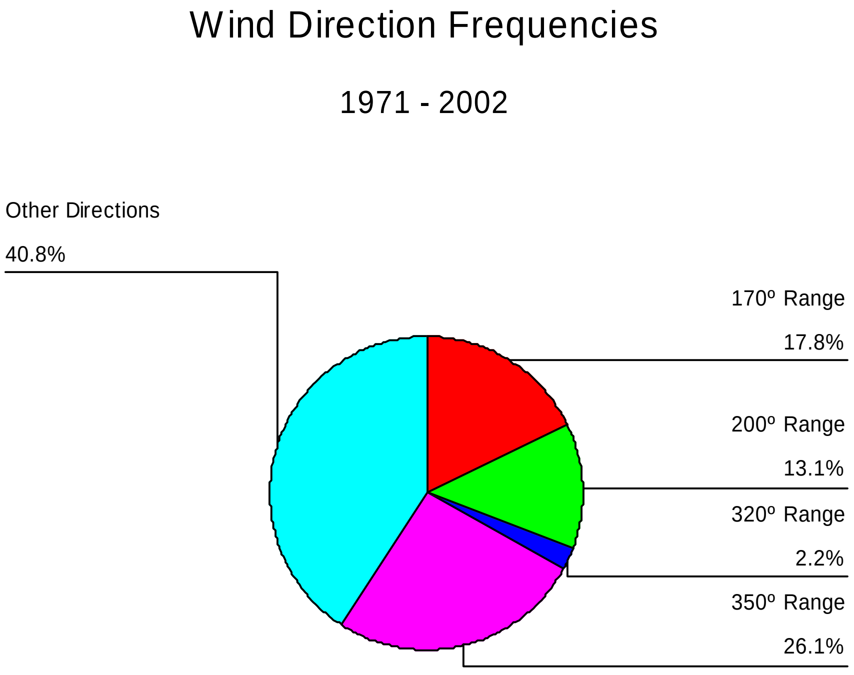


Figure 5 – Wind Direction Frequencies

It can be seen that around forty percent of all cases are excluded when the direction ranges are used. The directions tested in the wind tunnel are chosen because they represent the directions from which the strongest (and therefore most dangerous) Wellington winds blow. This suggests that by only selecting winds within these ranges, wind speeds will be generally higher than if all directions were considered.

The plot (Figure 6, next page) shows the distribution of directions of winds greater than 14 m/s. It is this plot that is used to justify the wind tunnel test directions, as wind tunnel tests are primarily concerned with identifying areas in the city with dangerous wind speeds i.e. wind speeds greater than 14 m/s.

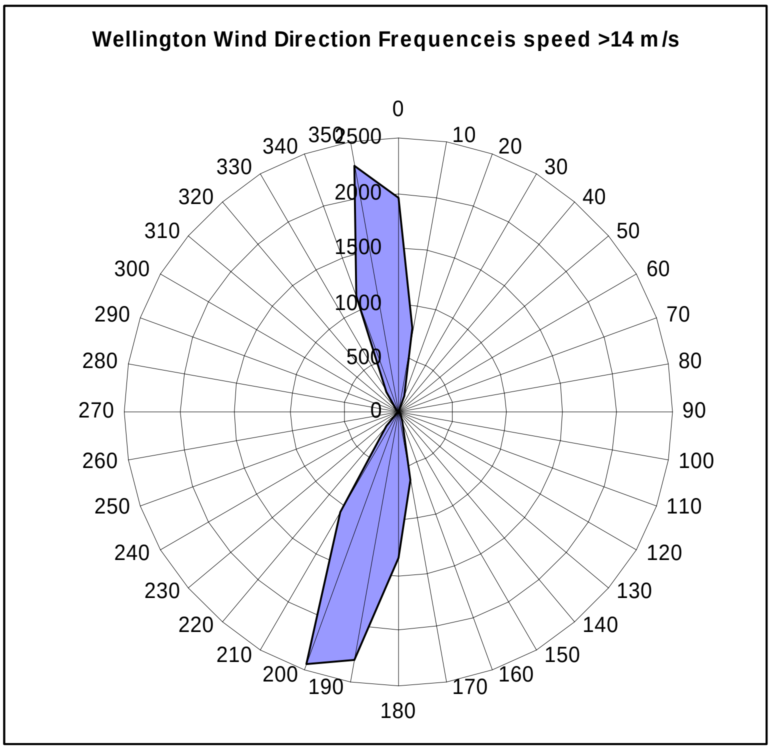
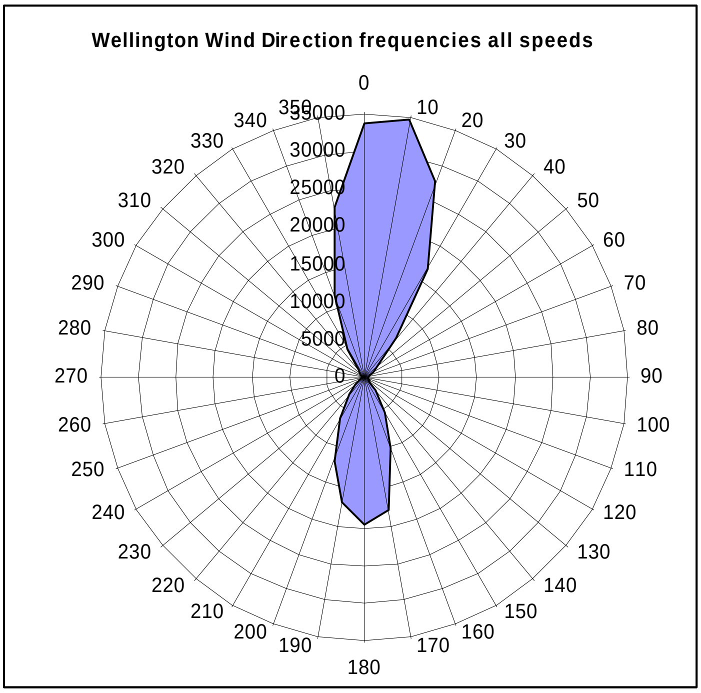


Figure 6 - Direction Frequencies for Wind Speeds > 14m/s

The next plot (Figure 7) shows the distribution of directions of winds of all speeds. It can be seen that most of the forty percent of excluded hourly directions come from the 0º - 30º sector. This suggests that if wind tunnel test were to be carried out for the sole purpose of urban quality assessment, they should consider directions other than the four that are currently tested.



Once these calculations have been performed, the data file has maximum and average wind columns for each point.

Now that local wind speeds are known, the next step is to calculate the other hourly parameters required for the globe thermometer temperature formula.

3.3 Solar & Globe Calculations

The globe thermometer formula requires the solar altitude to be calculated at each hour. The solar altitude angle describes the height of the sun at any hour, and is used to calculate the direct, diffuse, and reflected radiation values from measured total solar radiation. In order to calculate the solar altitude, the declination of the earth must be known. The declination represents the tilt of the earth, and can be calculated with a simple formula.

When these parameters are known, the four globe thermometer calculations may be performed. There are is a globe formula for wind speeds are less than 0.21 m/s and another for wind speeds are greater than 0.21 m/s. There are also “shaded” and “exposed” globe formulas. For each point four calculations with “if” statements need to be performed, and a further two calculations are needed to condense the data into two columns.

For each point following hourly variables are known; temperature (assumed to be constant across all points), maximum wind, average wind, shaded globe, exposed globe. It is now possible to perform the scale score calculation by applying the OPUS equation (1).

Opus Score Equation
= 26.581 + (Average wind * 10.254) + (Maximum wind * -5.567)
+ (Ambient Temperature * 0.498) + (Temperature Exposed (globe) * 0.8) (1)

Because there are two globe temperatures for each hour (shaded and exposed), two score calculations need to be performed. These calculations essentially give a “shaded score” and an “exposed score” for each point.

In order to determine which score is to be used for the final analysis, one must know at what times of the day and year the point is in sun and shade.

4.0 Urban Score

From the CAD analysis (completed separately – see Section 2) it is possible to tell which points are in sun or shade for any given hour, on "seasonally representative" days. A CAD analysis for "Point A" (a test point in Midland Park) would result in the data shown below.

Month	Number
Dec	12
Jan	1
Feb	2
Mar	3
Apr	4
May	5
Jun	6
Jul	7
Aug	8
Sep	9
Oct	10
Nov	11

Day	Hour	Sun / Shade
Jan-21	10	shade
	12	sun
	15	sun
Apr-21	10	shade
	12	sun
	15	shade
Jul-21	10	shade
	12	shade
	15	shade
Oct-21	10	shade
	12	sun
	15	shade

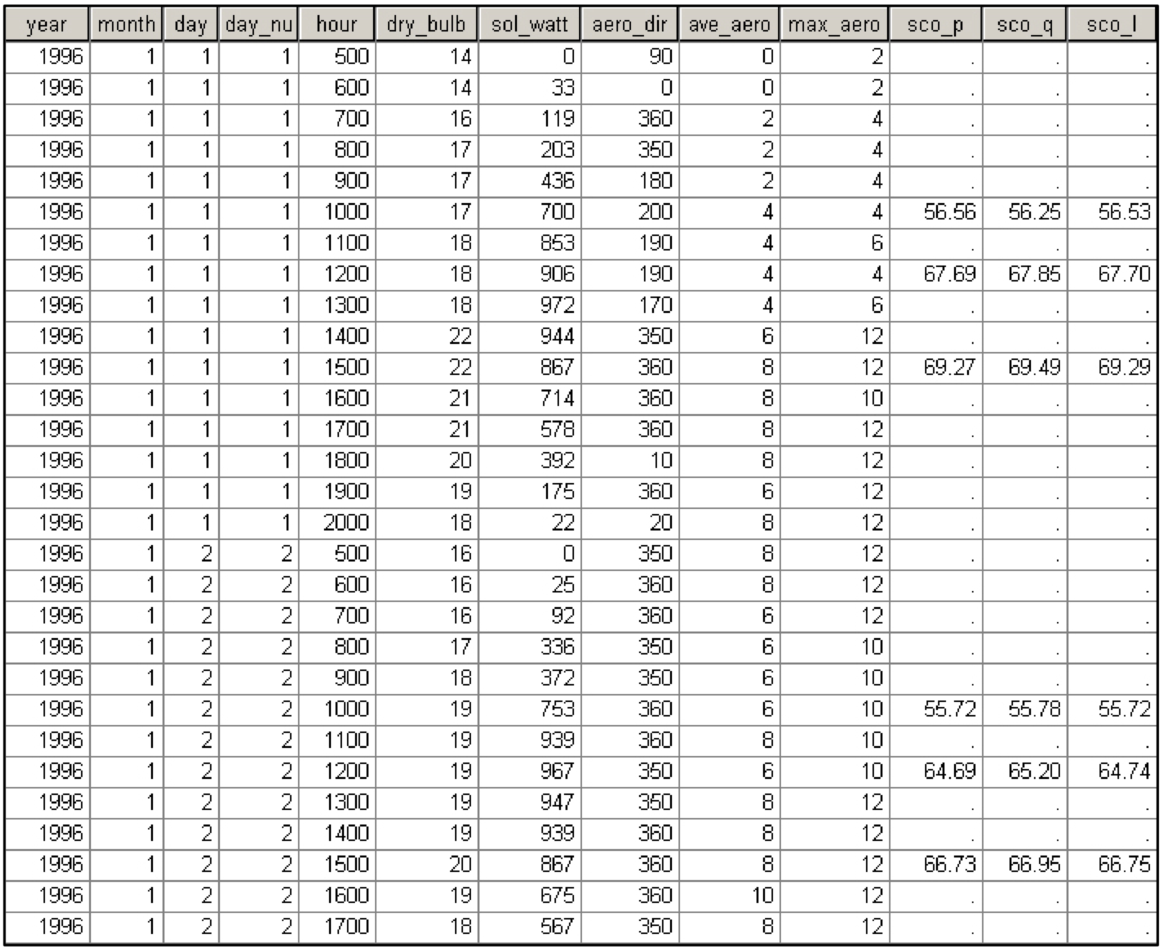
Figure 8 - Point A Analysis Example

It can be seen that on January 21st (a day chosen to represent December, January, and February) at 12 am, the "Point A" is in the sun.

Now that we have this data it is possible to write a series of "if" statements which select the appropriate score to use for different times of the day and year for the "Point A" analysis. For each hour being examined, there are 4 calculations with "if' statements.

This process will result in a single column of data which represents the scale score for one point in an urban space. The column will only have values for hours which have been analysed. Figure 9 (next page) is en example of a data which resulted from analysing three points (p, q, l) at three times of the day (10, 12, 3).

This data is now ready to be statistically analysed.



year	month	day	day_nu	hour	dry_bulb	sol_watt	aero_dir	ave_aero	max_aero	sco_p	sco_q	sco_l
1996	1	1	1	500	14	0	90	0	2	.	.	.
1996	1	1	1	600	14	33	0	0	2	.	.	.
1996	1	1	1	700	16	119	360	2	4	.	.	.
1996	1	1	1	800	17	203	350	2	4	.	.	.
1996	1	1	1	900	17	436	180	2	4	.	.	.
1996	1	1	1	1000	17	700	200	4	4	56.56	56.25	56.53
1996	1	1	1	1100	18	853	190	4	6	.	.	.
1996	1	1	1	1200	18	906	190	4	4	67.69	67.85	67.70
1996	1	1	1	1300	18	972	170	4	6	.	.	.
1996	1	1	1	1400	22	944	350	6	12	.	.	.
1996	1	1	1	1500	22	867	360	8	12	69.27	69.49	69.29
1996	1	1	1	1600	21	714	360	8	10	.	.	.
1996	1	1	1	1700	21	578	360	8	12	.	.	.
1996	1	1	1	1800	20	392	10	8	12	.	.	.
1996	1	1	1	1900	19	175	360	6	12	.	.	.
1996	1	1	1	2000	18	22	20	8	12	.	.	.
1996	1	2	2	500	16	0	350	8	12	.	.	.
1996	1	2	2	600	16	25	360	8	12	.	.	.
1996	1	2	2	700	16	92	360	6	12	.	.	.
1996	1	2	2	800	17	336	350	6	10	.	.	.
1996	1	2	2	900	18	372	350	6	10	.	.	.
1996	1	2	2	1000	19	753	360	6	10	55.72	55.78	55.72
1996	1	2	2	1100	19	939	360	8	10	.	.	.
1996	1	2	2	1200	19	967	350	6	10	64.69	65.20	64.74
1996	1	2	2	1300	19	947	350	8	12	.	.	.
1996	1	2	2	1400	19	939	360	8	12	.	.	.
1996	1	2	2	1500	20	867	360	8	12	66.73	66.95	66.75
1996	1	2	2	1600	19	675	360	10	12	.	.	.
1996	1	2	2	1700	18	567	350	8	12	.	.	.

Figure 9 - Analysed SPSS Example data

4.1 Variable Input Values

The input variables for the scale score equation can be derived two main ways. What follows is a description of each of the methods, and a brief discussion of their advantages, disadvantages, and possible inaccuracies. The main differences between the methods are the way in which they deal with sun and wind data.

4.1.1 Method One

The first method uses the whole space (in this case Midland Park) for solar analysis. The CAD analysis results in the data table shown below (Figure 10, next page). The figures simply show the percentage of the park in sun and shade, for different times of the day, on different days of the year. This type of solar analysis is accurate; however some approximation does occur when using "typical seasonal days".

	10 am		12 pm		3 pm	
	Sun	Shade	Sun	Shade	Sun	Shade
Jan 21	15%	85%	76%	24%	82%	18%
April 21	0%	100%	13%	87%	70%	30%
July 21	0%	100%	0%	100%	0%	100%
Oct 21	0%	100%	59%	41%	0%	100%

Figure 10 - Sun/Shade Ratios

With regard to wind, this method uses an "average" point in the park for analysis. In the Midland Park example a point in the centre of the park was chosen. This point was chosen as it was very close to the point where OPUS took environmental measurements.

It must be noted that the way in which the "average" point is selected can vary, according to the data available. For example, if wind ratios at several different points in an urban space are known, the mean ratio can be used.

Because only one wind ratio is considered, it is relatively simple to arrive at a final score. At the single point both the shaded and exposed scores are calculated. The scores are then weighted according to the sun/shade ratio for a particular time of the day or year. For example on April 21 (or any day in March, April, of June) the final score would be calculated by computing 0.13 * exposed score + 0.87 * shaded score.

This method results in a single number score for an urban space, which can provide a quick indication of the conditions in the park. Using this type of score can however mean that important information, such as good or bad areas in the park, is lost.

A sample output from an analysis of Midland Park can be seen in Figure 11.

Mean Score

	12	12	3
Whole park sun analysis, center point wind ratio	49.49	53.14	52.51
Whole park sun analysis, mean wind ratio	49.58	53.30	52.62

Figure 11 - Method One Output

4.1.2 Method Two

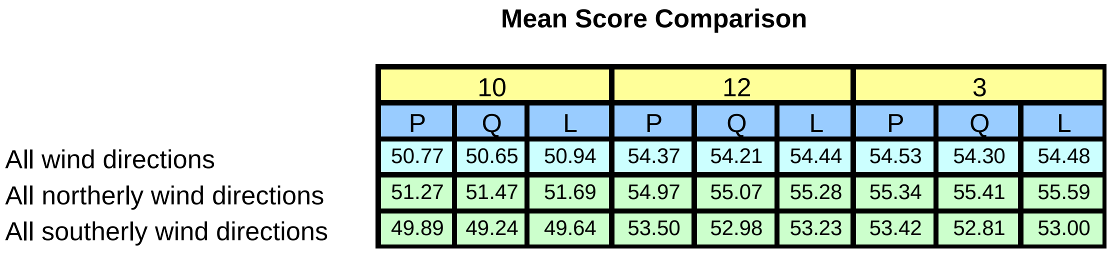
This method uses wind tunnel measurement points as the basis of urban space analysis. At these points the wind ratios for each wind direction are known, and consequently the wind speed at each of these points can be calculated very accurately.

In terms of solar analysis, each point is examined at the desired hours during the "typical seasonal days". The analysis simply consists of determining weather the point is in sun or shade. From there the appropriate score (either exposed or shaded) is selected.

This output from this method consists of scores at the desired hours of the day, for the points that were considered. It must be noted that the accuracy of this method is dependant on the number of points. The distribution of the points within the space is also important, if for example the points are clustered in a sunny area of a part of the park, the score may suggest that conditions in the park are better than they really are.

Ideally a grid of points should be tested, however most wind tunnel tests only take speed measurements at points were there are foreseeable problems. This means that most points that are tested are likely to be the places where the wind environment is worst. In order to overcome this bias, wind tunnel measurements could be done at regular intervals.

This method retains important information about the space. Because individual points are considered, good and bad areas in an urban space may be identified. Figure 12 illustrates the information rich output that method two yields.



Mean Score Comparison

	10			12			3		
	P	Q	L	P	Q	L	P	Q	L
All wind directions	50.77	50.65	50.94	54.37	54.21	54.44	54.53	54.30	54.48
All northerly wind directions	51.27	51.47	51.69	54.97	55.07	55.28	55.34	55.41	55.59
All southerly wind directions	49.89	49.24	49.64	53.50	52.98	53.23	53.42	52.81	53.00

Figure 12 - Method Two Output

6.0 Bibliography

AutoCAD Help: User Documentation

Locke, Nick, *Erosion Contours and Hotfilm Wind Speeds*, Central Laboratories Report

7.0 Appendix 1

7.1 AutoLISP Solrender Routine

```
; ==============================================================
;
;              START OF SOLRENDER.LSP
;
; ==============================================================

(command  ^C  ^C)

(print "loading solrender lisp...")

; ================================================
; Dialog box that explains the lisp when loaded.
; ================================================
(defun c:solhelp ()
  (alert
"________________________________

      SOLAR SHADING ANALYSIS TOOL
________________________________

This AutoLISP Renders hourly from one time
to another time for a day you specify.

The renderings are saved as tiff's in
the folder of your drawing with the name
as the hour that they represent.

This lisp assumes that no lights exist
when it is run. It creates and deletes
a light called temp_dis when running,
which may remain and cause errors if
lisp is canceled while rendering. To
fix just delete the light.
________________________________

              COMMANDS:
________________________________

    SOLRENDER
     To Run

    SETSOLRENDER
     Sets:
     - image size
     - output location
     - global position
     - hourly/half-hourly renderings

    SOLDEFAULTS
     Resets variables to default

    SOLHELP
     This dialog box
________________________________

         CHRIS FURNEAUX 2002
________________________________")
  )

(c:solhelp)

; Other commands:
;
; c:set_sol_imagesize          : sets output image size
; set_sol_location             : sets location of model
; c:set_sol_halfhours               : sets if half hour rendering is to be included
; c:set_sol_dir                              : sets directory to render to
; c:sol_setmonthend (month)         : sets the end day of the month in the variable monthendday
; c:sol_setrenderer                 : sets up rendering parameters
; c:sol_setptime                    : sets the string for the output file name
; c:sol_checkinputs                 : checks that location and image size are set
; error (message)                   : writes an error message and exits task
; c:sol_setinputs                   : sets inputs
; c:sol_solrender                   : runs render with conditions set when run
; c:sol_renderloop                  : runs rendering loop


; ==========================================
; Sets start (default) values for the lisp.
; edit this to change defaults.
; ==========================================

(defun c:soldefaults ()
        (setq fileformat "TIFF"); file format of rendered image
        (setq xres 300)                         ; x resolution of image
        (setq yres 300)                         ; y resolution of image
        (setq aratio 1)                         ; aspect ratio of image
        (setq colormode "C24")                  ; colourmode of image
        (setq halfhours "N")            ; weather to render on the half hour as well by default
        (setq sollat -41.28)            ; default latitude
        (setq sollong 174.77)           ; default longitude
        (setq soltime -12)              ; default timezone
        (setq soldir "")                ; directory to render to.
        (print "solrender defaults set...")
  )

(c:soldefaults)

(arxload "acrender.arx" "unable to load acrender.arx : please open and close the light dialog and run this again")

; ==============================
; Functions to change settings.
; ==============================

(defun c:setsolrender ()
  (c:set_sol_imagesize)
  (set_sol_location)
  (c:set_sol_halfhours)
  (c:set_sol_dir)
  )

(defun c:set_sol_imagesize ()       ; Sets the image size
  (setq setsolimgsze "N")
  (initget 1 "Y N")
  (setq setsolimgsze (getkword "Set output image size? (Y/N)"))
         (if (= setsolimgsze "Y")
                (progn
                  (while (< xres 0)
                         (setq xres (getint "enter the x resolution of the image:"))
                    )
                  (while (< yres 0)
                         (setq yres (getint "enter the y resolution of the image:"))
                    )
                )
          )
  )

(defun c:set_sol_location ()        ; Sets the location of the rendering
  (setq setsollocation "N")
  (initget 1 "Y N")
  (setq setsollocation (getkword "Set latitude, longitude and timezone? (Y/N)"))
         (if (= setsollocation "Y")
                (progn
                  (setq sollat 181)
                  (while (> sollat 180)
                    (setq sollat -181)
                    (while (< sollat -180)
                         (setq sollat (getreal "enter the latitude (-180 to 180):"))
                 )
                    )
                  (setq sollong 181)
                  (while (> sollong 180)
                    (setq sollong -181)
                    (while (< sollong -180)
                         (setq sollong (getreal "enter the longitude (-180 to 180):"))
                 )
                    )
                  (setq sollong 14)
                  (while (> soltime 12)
                    (setq soltime -14)
                    (while (< soltime -12)
                         (setq soltime (getreal "enter the timezone (-12 to 12):"))
                 )
                    )
                )
          )
  )

(defun c:set_sol_halfhours ()       ; Sets weather renderings are done on the hour or the halfhour
  (setq halfhours "Y")
  (initget 1 "Y N")
  (setq halfhours (getkword "Do you want renderings on half hours (Y/N)"))
  )

(defun c:set_sol_dir ()
  ; Sets the location to save the image to
  (setq setsoldir "N")
  (initget 1 "Y N")
  (setq setsoldir (getkword "Do you want to set the directory to save images to? (Y/N)"))
         (if (= setsoldir "Y")
          (setq soldir (getstring "Enter location for images (e.g. drive:\full_path\): "))
          )
  )

; =========================================
; Functions used by the rendering routine.
; =========================================

; ===================
; Set months end day.
; ===================

(defun c:sol_setmonthend (monthvar)
  (graphscr)
        (if (= monthvar 1) (setq monthendday 31)) ; Jan
        (if (= monthvar 2) (setq monthendday 28)) ; Feb
        (if (= monthvar 3) (setq monthendday 31)) ; Mar
        (if (= monthvar 4) (setq monthendday 30)) ; Apr
        (if (= monthvar 5) (setq monthendday 31)) ; May
        (if (= monthvar 6) (setq monthendday 30)) ; Jun
        (if (= monthvar 7) (setq monthendday 31)) ; Jul
        (if (= monthvar 8) (setq monthendday 31)) ; Aug
        (if (= monthvar 9) (setq monthendday 30)) ; Sep
        (if (= monthvar 10) (setq monthendday 31))            ; Oct
        (if (= monthvar 11) (setq monthendday 30))            ; Nov
        (if (= monthvar 12) (setq monthendday 31))            ; Dec
  )


; ================
; Set up Renderer.
; ================

(defun c:sol_setrenderer ()
  (graphscr)

        ; ============================================================
        ; Loads the acrender.arx that is needed to enable renderings.
        ; ============================================================
        (arxload "acrender.arx" "***error*** unable to load acrender.arx : please open and close the light dialog and run this again")

        ; ==========================
        ; Sets up rendering options.
        ; ==========================

        (c:rfileopt fileformat xres yres aratio colormode)
        (c:rpref "DEST" "FILE")
        (c:rpref "STYPE" "ARAY")
        (c:rpref "SELECT" "ALL")
        (c:rpref "TOGGLE" "SHADOW" "ON")
        (c:rpref "TOGGLE" "CACHE" "ON")
        (c:rpref "TOGGLE" "SHADOW" "ON")
        (c:rpref "TOGGLE" "FINISH" "ON")
  )


; ===================
; Set up print string.
; ===================

(defun c:sol_setptime ()
 (graphscr)
        (if (< ptime 1000)
         (progn
          (if (< day 10)
               (setq ptimedate (strcat soldir "m" (vl-princ-to-string month) "_d0" (vl-princ-to-string day) "_h0" (vl-princ-to-string ptime)))
               (setq ptimedate (strcat soldir "m" (vl-princ-to-string month) "_d" (vl-princ-to-string day) "_h0" (vl-princ-to-string ptime)))
           )
          )
         (progn
          (if (< day 10)
               (setq ptimedate (strcat soldir "m" (vl-princ-to-string month) "_d0" (vl-princ-to-string day) "_h" (vl-princ-to-string ptime)))
               (setq ptimedate (strcat soldir "m" (vl-princ-to-string month) "_d" (vl-princ-to-string day) "_h" (vl-princ-to-string ptime)))
           )
          )
         )
 )


; =========================================
; Checks that inputs are safe (error control).
; =========================================

(defun c:sol_checkinputs ()
 (graphscr)
  (if (or (< xres 1) (< yres 1)) (error "Image size"))
  (if (or (< sollat -180) (> sollat 180)) (error "Latitude"))
  (if (or (> sollong 180) (< sollong -180)) (error "Longitude"))
  (if (or (< soltime -12) (> soltime 12)) (error "Timezone"))
 )


; ========================================
; bad input quit with error message function.
; ========================================

(defun error (msg)
 (princ (strcat "***error***: Bad " msg ", Please set. (SOLRENDERSET/SOLRENDERDEFAULT)"))
 (princ "
")
 (exit)
)


; ===================================
; Prompt for rendering times and dates.
; ===================================

(defun c:sol_setinputs ()
 (graphscr)
```

```
  ; ***sets the times of the day to be rendered for each day specified***
  (setq sttime 0)
  (setq fintime 0)

  (while (or (< sttime 1) (> sttime 24))
        (setq sttime (getint "what time should the renderings start?(hour in 24HR time):"))
   )
  (while (or (> sttime fintime) (< fintime 1) (> fintime 24))
        (setq fintime (getint "what time should the renderings finish?(hour in 24HR time):"))
   )
  (setq fintime (+ fintime 1))

  (initget 1 "Y N")
  (setq solmultiple (getkword "Do you want to render more then one day? (Y/N)"))

  ; ***for setting a single day for rendering***
  (if (= solmultiple "N")
        (progn
         (setq stmonth 0)
         (while (or (< stmonth 1) (> stmonth 12))
                (setq stmonth (getint "what month of the year (1-12):"))
          )
         (c:sol_setmonthend stmonth)
         (setq stday 0)
         (while (or (< stday 1) (> stday monthendday))
                (setq stday (getint "what day of that month?"))
          )
        (setq finmonth stmonth)
        (setq finday stday)
        )
   )

  ; ***for setting a period of days to be rendered on***
  (if (= solmultiple "Y")
        (progn

         ; ***Sets the starting month and day***
         (setq stmonth 0)
         (while (or (< stmonth 1) (> stmonth 12))
                (setq stmonth (getint "what month of the year to start?:"))
          )
         (c:sol_setmonthend stmonth)
         (setq stday 0)
         (while (or (< stday 1) (> stday monthendday))
                (setq stday (getint "what day of that month?"))
           )

         ; ***Sets the finishing month and day***
         (setq finmonth 0)
         (while (or (< finmonth 1) (> finmonth 12) (< finmonth stmonth))
                (setq finmonth (getint "what month of the year to finish?:"))
          )
         (c:sol_setmonthend finmonth)
         (if (= stmonth finmonth) (setq monthstday stday) (setq monthstday 1))
         (setq finday 0)
         (while (or (< finday 1) (> finday monthendday) (< finday monthstday))
                (setq finday (getint "what day of that month?"))
          )
        )
   )
 )


; ========================================
; Rendering task that is run every hour.
; ========================================

(defun c:sol_solrender ()
 (graphscr)

 ; ***sets the filename to be used for image***
 (setq ptime (* 100 time))
```

```
  (c:sol_setptime)
  ; ***creates a light for the specified time***
  (c:light "ND" "temp_dis" 1 nil nil '(1 1 1) 0 nil nil 0 "on" month day time 0 "off" sollat sollong soltime)
  ; ***renders an image to the specified filename***
  (c:render (vl-princ-to-string ptimedate))
  ; ***deletes the light***
  (c:light "D" "temp_dis")

  ; ==========================================================
  ; Runs Renderings on the halfhours when specified to do so.
  ; ==========================================================

  (if (= halfhours "Y")
   (progn
    ; ***sets the filename to be used for image***
    (setq ptime (+ 30 ptime))
    (c:sol_setptime)
    ; ***creates a light for the specified time***
    (c:light "ND" "temp_dis" 1 nil nil '(1 1 1) 0 nil nil 0 "on" month day time 30 "off" sollat sollong soltime)
    ; ***renders an image to the specified filename***
    (c:render (vl-princ-to-string ptimedate))
    ; ***deletes the light***
    (c:light "D" "temp_dis")
    )
   )

   ; =========================
   ; End halfhour rendering.
   ; =========================
  )

; =================
; Rendering loops.
; =================

(defun c:sol_renderloop ()
  (graphscr)
  (setq month stmonth)
  (setq finmonth (+ finmonth 1))
  (while (< month finmonth)       ; Start month loop.
   (if (= stmonth month) (setq day stday) (setq day 1))
   (if (= finmonth (+ month 1))
    (setq thisfinday finday)
    (progn
         (c:sol_setmonthend month)
         (setq thisfinday monthendday)
         )
    )                                      ; reset current day to time to start day for month
   (Setq thisfinday (+ thisfinday 1))
   (while (< day thisfinday)       ; Start day loop.
    (setq time sttime)             ; reset current time to time to start time
    (while (< time fintime)        ; Start hour loop.
         (c:sol_solrender)         ; run the renders
         (setq time (+ time 1)); step the loop foward one time step.
         )                                 ; End hour loop.
    (setq day (+ day 1))           ; step the loop foward one time step.
    )                              ; end day loop.
   (setq month (+ month 1))        ; step the loop foward one time step.
   )                                       ; end month loop.
   (princ)                                 ;exit quietly
  )


; ========================================
; Function to run the rendering routine.
; ========================================

(defun c:solrender ()
  (graphscr)

  (c:sol_checkinputs)
  (c:sol_setrenderer)
  (c:sol_setinputs)
  (c:sol_renderloop)

)                                        ; End solrender function.


(print "***solrender lisp loaded***")

; ============================================================
;
;                END OF SOLRENDER.LSP
;
; ============================================================
```

8.0 Appendix 2

8.1 Tiff to Text File Conversion

To convert a tiff image into a text file that can be imported into Excel:

Batch file:

Tif2txt.bat

- Converts a tiff into a ASCII data array showing x,y,r,g,b
- output is 5 columns of numbers (X,Y,R,G,B)
- X&Y are pixel co-ordinates
- RGB values are given as 0-2.55
- input tif needs to be a rgb or grayscale tif format (not indexed)
- useage: tif2txt input_1 input_2 ..... input_n
- needs ra_tiff.exe and pvalue.exe to run (should be accessible from anywhere on any computer with rayfront installed (via the default path)

Batch file by Chris Furneaux.

------------------------------ADDITIONAL INFO------------------------------------

The files ra_tiff.exe and pvalue.exe and can be found in the folder on any computer where rayfront is installed.

Use ra_tiff in reverse (-r) to turn a .tiff into a .pic (radiance picture) then use pvalue on the .pic

Rayfront (homepage: http://www.schorsch.com) is a user interface for Radiance (homepage: http://radsite.lbl.gov/radiance/HOME.html)

A manual page for ra_tiff can be found here:
http://www.schorsch.com/rayfront/manual/htmlman/ra_tiff.html

pvalue can be found here:
http://www.schorsch.com/rayfront/manual/htmlman/pvalue.html

```
####################################
Command lines:
ra_tiff -r input.tif output.pic
pvalue output.pic > output.txt
####################################
```

To run:

1. Make sure the tiff file to be converted is in the same file as the convert package (which can be unzipped from tif2txt.zip).

2. The tiff file cannot have any Alpha Channels, you may need to check this in Photoshop.

3. Right-click on runme.bat and edit the file using notedpad/wordpad. Replace (tiff filename) with the name of the file to be converted. For example if the file to be converted is called test.tif, the batch file should read:

```
call tif2txt test
```

If you want to process more than one file at a time, then just copy the first line of the runme.bat file and change the name of the file to be processed, for example if you had files called: test.tif, test1.tif and test2.tif the batch file should read:

```
call tif2txt test
call tif2txt test1
call tif2txt test2
```

4. Save the runme.bat file and close it down.

5. Double click on runme.bat and it will run, it may bring up a couple of line errors, but just click OK to ignore them. Your txt files should appear in the tif2txt folder.

9.0 Appendix 3

9.1 Limitations of the Bead Erosion Plots

(Paul Carpenter, Aerodynamics Section, Central Labs, Ministry of Works and Development, Lower Hutt)

1. Vertical Velocity Components - beads more easily distributed where there is a downward component in the air velocity, than where the flow is horizontal.
2. Vortices - beads can accumulate in the core of a steady vortex. This area can be seen to be relatively calm, because the vortex is not visible.
3. Bead Mounds - beads tend to accumulate in mounds as each test proceeds, mounds subsequently modify the flow conditions in their vicinity.

Maps have the potential to sustain conditions indicated. Hence, speeds indicated are not necessarily a constant occurrence.

Photos taken during wind speed tests are taken at the same time.

Flow visualisation tests (beads):
- indication of relative degree of windiness adjacent to the building and surrounding streets.
- clearance of beads - magnitude etc., indicates a windy area
- wind speed measurements made at points indicated by the bead clearance
- done in both a Northerly and a Southerly

10.0 Appendix 4

10.1 Wind Tunnel Bead Erosion Contours and Hotfilm Wind Speeds

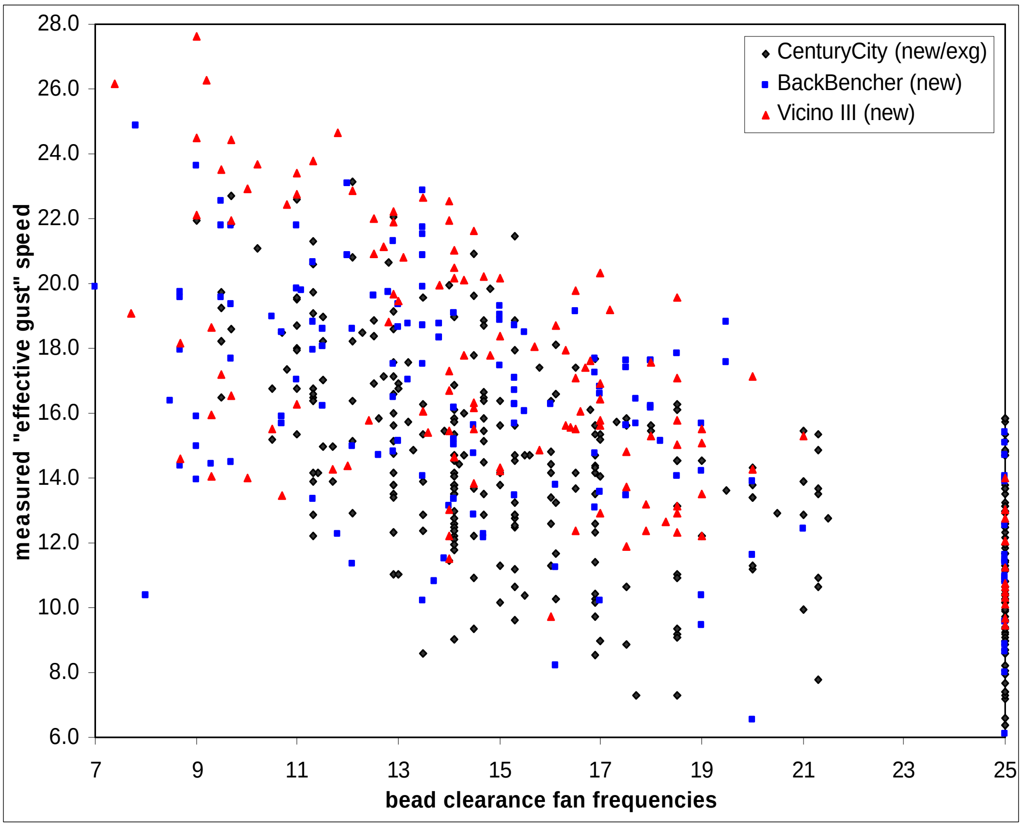
Bead erosion contours from three wind tunnel studies (Century City – 70 Tory Street, Backbencher Apartments – Cnr. Molesworth & Kate Sheppard Place, Vicino Apartments, Stage III – 29 Taranaki Street) have been analysed with a view to determining their suitability for measuring pedestrian level wind speeds that are currently measured using a hotfilm probe.

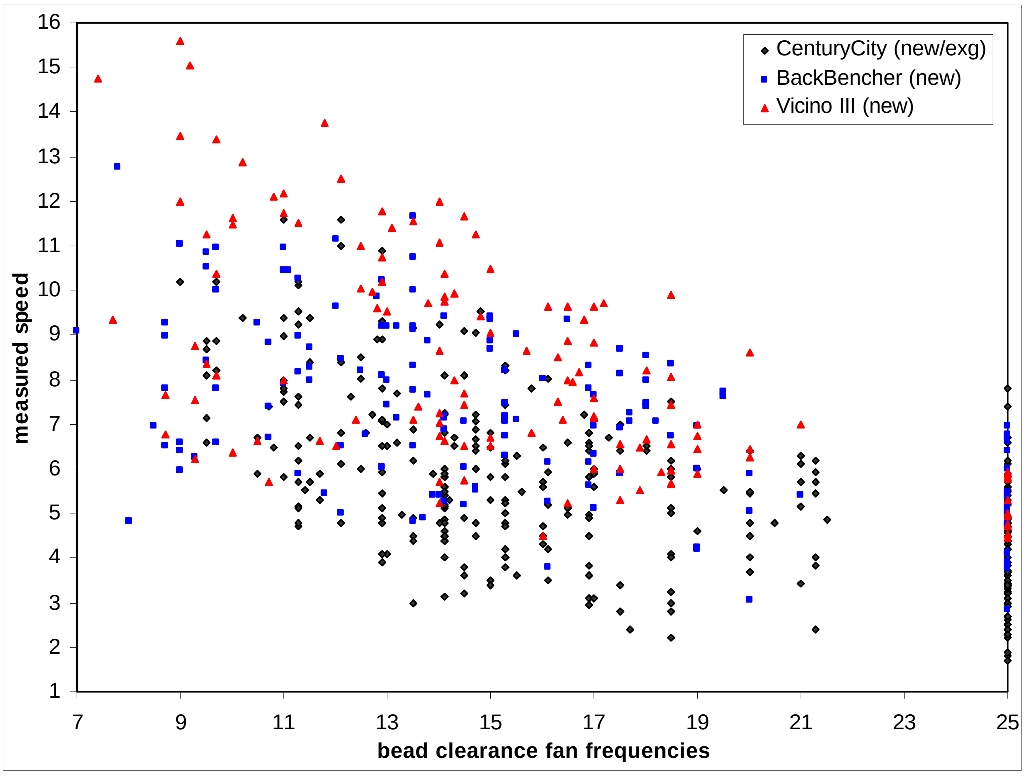
Contours were produced using scanned photos from the standard bead tests that we currently perform. “Photoshop” was used to process the images (draw the contour lines and combine them into one image). This process was fairly ‘automatic’ with little scope to influence, or mess up, the contours produced. Each contour corresponds to the edge of the beads for a given wind tunnel speed. As the fan control frequency is proportional to the wind tunnel speed, I have assigned to each contour the frequency value that corresponds to the bead erosion pattern in each photograph. The table below is the key to these values.

Photograph	Fan Control Frequency (Hz)	Reference Tunnel Speed (m/s)	Colour (RGB)
0	0.0	0.0	Black (0 0 0)
1	7.7	1.7	Yellow (255 255 0)
2	9.7	2.2	Orange (255 125 0)
3	11.3	2.6	Red (255 0 0)
4	12.9	3.0	Magenta (200 0 255)
5	14.1	3.3	Blue (0 0 255)
6	15.3	3.6	Light blue (0 255 255)
7	16.9	4.0	Light green (0 255 0)
8	18.5	4.4	Dark green (0 125 0)
9	21.3	5.1	Brown (100 50 0)

The least accurate part of the analysis was assigning ‘frequencies’ to the locations where wind speeds were measured with the hotfilm probe. Two problems arose here - 1) identifying the location on the image and 2) interpolating a frequency value from the contours. I made a rough estimate of the potential error associated with each value. Where a location coincided with no bead erosion (even at the highest wind speed) a frequency value of 25 was assigned to that point.

What follows are summary plots of the ‘bead clearance fan frequency’ versus hotfilm wind speeds for all three wind tunnel studies.


28.0
26.0
24.0
22.0
20.0
18.0
16.0
14.0
12.0
10.0
8.0
6.0
measured "effective gust" speed
7
9
11
13
15
17
19
21
23
25
bead clearance fan frequencies
◆ CenturyCity (new/exg)
■ BackBencher (new)
▲ Vicino III (new)


16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
measured speed
7
9
11
13
15
17
19
21
23
25
bead clearance fan frequencies
◆ CenturyCity (new/exg)
■ BackBencher (new)
▲ Vicino III (new)