NAME

alloc_l1 – allocate memory for level-1 record

SYNOPSIS

int alloc_l1 (int *npix***, l1str ***_l1rec_**);**

ARGUMENTS

npix	number of pixels
l1rec	level-1 data record

DESCRIPTION

allocates 1 level-1b record to hold data for a single scan of "npix" pixels.

RETURNS

0 if failed, 1 if successful

CALLED BY

alloc_l1q

NAME

alloc_l2 – allocate memory for level-2 record

SYNOPSIS

int alloc_l2 (int *npix*, **l2str *** *l2rec*);

ARGUMENTS

npix	number of pixels
l2rec	level-2 data record

DESCRIPTION

allocates 1 level-2 record to hold data for a single scan of "npix" pixels.

RETURNS

0 if failed or number of bytes allocated

CALLED BY

main

NAME

alloc_target – allocate memory for target record

SYNOPSIS

int alloc_target (int *npix*, **tgstr ***_rec_**);**

ARGUMENTS

npix	number of pixels
rec	target data record

DESCRIPTION

allocates 1 target record to hold data for a single scan of "npix" pixels.

RETURNS

0 if failed, or number of bytes allocated

CALLED BY

main

NAME

b128_box_num – find index for landmask

SYNOPSIS

int b128_box_num (float *lat*, **float** *lon*, **float ***_lat_off_, **float ***_lon_off_**);**

ARGUMENTS

lat	latitude of point from -90. to 90. (deg)
lon	longitude of point from -180 to 180. (deg)
lat_off	new positive latitude relative to mask (0.0 to 180.0) (deg)
lon_off	new positive longitude relative to mask (0.0 to 360.0) (deg)

DESCRIPTION

for the 128 / degree land mask file, find the index of the degree box from the lat, lon

RETURNS

the index (0-rel) of the 1 degree box in mask

CALLED BY

b128_msk_get

AUTHOR

W. Robinson 8-Oct-1997 Original development

NAME

b128_msk_get – get mask for given location

SYNOPSIS

int b128_msk_get (float *lat*, **float** *lon*, **int** *msknum*);

ARGUMENTS

lat	latitude of point from -90. to 90. (deg)
lon	longitudeof point from -180 to 180. (deg)
msknum	which mask to use (0 or 1)

DESCRIPTION

For an incoming latitude, longitude find if it is over mask or not using the 128 by 128 / degree mask file

RETURNS

1 for mask on, 0 for mask off, -1 for problem

CALLED BY

setflags

AUTHOR

W. Robinson	8-Oct-1997	Original development
W. Robinson	19-Dec-1997	adapt to use 2 mask files

NAME

b128_msk_init – initial landmask

SYNOPSIS

int b128_msk_init (char ***_landfile_, int** _msknum_**);**

ARGUMENTS

landfile	name of mask file to use
msknum	number of the mask (0 or 1)

DESCRIPTION

initialize the 218 x 128 per degree resolution processing to get the a mask. This code is designed to contain two identical masks. One is the landmask and the other is the bathymetery (shallow water massk).

RETURNS

0 is good, else an open or read error

CALLED BY

setflags

AUTHOR

W. Robinson	8-Oct-1997	Original development
W. Robinson	19-Dec-1997	adapt to use 2 mask files

NAME

b128_wd_bit – get word, bit location for location

SYNOPSIS

int b128_wd_bit (float *lat_off* **, float** *lon_off* **, int ***_box_wd_**, int ***_box_bit_**);**

ARGUMENTS

lat_off	-- undescribed --
lon_off	-- undescribed --
box_wd	word # in array of 128 sq box to find desired mask value
box_bit	bit # in word (*box_wd*) of desired mask value

DESCRIPTION

Get the word and bit number in said word in array

CALLED BY

b128_msk_get,

AUTHOR

W. Robinson 8-Oct-1997 Original development

NAME

carder_init – initialize Carder algorithms

SYNOPSIS

int carder_init (int *sensorID*, **int** *carder_opt*, **int** *carder_version*, **int** *carder_iter*, **float** **carder_gains*);

ARGUMENTS

sensorID	sensor
carder_opt	selects desired carder packaging algorithm
carder_version	selects desired carder algorithm
carder_iter	turns on iteration of the s term
carder_gains	gain adjustments to rrs spectrum for carder algorithm only

DESCRIPTION

The **carder_init** routine is used to initialize the Carder processing algorithms. The routine reads in the appropriate input parameters file from $AUTO_DATA/

CALLED BY

main

NAME

carder_v1_3 – run carder's algorithm version 1.3

SYNOPSIS

int carder_v1_3 (double ****rrs*, double ****atotPtr***, double ****aphPtr***, double ****adgPtr***, double ****bbPtr***, double ****chlPtr***, double ****agdefPtr***, double ****aphdefPtr***, double ****agmodPtr***, double ****aphmodPtr***, double ****sPtr***, int** *iter_s***, short *** *flags***);**

ARGUMENTS

rrs	remote sensing reflectance in nm for first six SeaWiFS wavelengths
atotPtr	output total absorption spectrum (1/m)
aphPtr	output phytoplankton absorption spectrum (1/m)
adgPtr	output CDOM absorption spectrum (1/m)
bbPtr	output backscattering spectrum (1/m)
chlPtr	output chlorophyl-a (mg/m^3)
agdefPtr	unused
aphdefPtr	unused
agmodPtr	unused
aphmodPtr	unused
sPtr	output spectral value
iter_s	iterate on s flag
flags	carder flags

NAME

carder_v1_33 – run carder's algorithm version 1.33

SYNOPSIS

int carder_v1_33 (double ***_rrs_, double ***_atotPtr_**, double ***_aphPtr_**, double ***_adgPtr_**, double ***_bbPtr_**, double ***_chlPtr_**, double ***_agdefPtr_**, double ***_aphdefPtr_**, double ***_agmodPtr_**, double ***_aphmodPtr_**, double ***_sPtr_**, int** _iter_s_**, short *** _flags_**);**

ARGUMENTS

rrs	remote sensing reflectance in nm for first six SeaWiFS wavelengths
atotPtr	output total absorption spectrum (1/m)
aphPtr	output phytoplankton absorption spectrum (1/m)
adgPtr	output CDOM absorption spectrum (1/m)
bbPtr	output backscattering spectrum (1/m)
chlPtr	output chlorophyl-a (mg/m^3)
agdefPtr	output default adg400
aphdefPtr	output default aph675
agmodPtr	output modeled adg400
aphmodPtr	output modeled aph675
sPtr	output spectral value
iter_s	iterate on s flag
flags	carder flags

NAME

carder_v1_4 – run carder's algorithm version 1.4

SYNOPSIS

int carder_v1_4 (double ***_rrs_, double ***_atotPtr_**, double ***_aphPtr_**, double ***_adgPtr_**, double ***_bbPtr_**, double ***_chlPtr_**, double ***_agdefPtr_**, double ***_aphdefPtr_**, double ***_agmodPtr_**, double ***_aphmodPtr_**, double ***_sPtr_**, int** _iter_s_**, short *** _flags_**);**

ARGUMENTS

rrs	remote sensing reflectance in nm for first six SeaWiFS wavelengths
atotPtr	output total absorption spectrum (1/m)
aphPtr	output phytoplankton absorption spectrum (1/m)
adgPtr	output CDOM absorption spectrum (1/m)
bbPtr	output backscattering spectrum (1/m)
chlPtr	output chlorophyl-a (mg/m^3)
agdefPtr	output default adg400
aphdefPtr	output default aph675
agmodPtr	output modeled adg400
aphmodPtr	output modeled aph675
sPtr	output spectral value
iter_s	iterate on s flag
flags	carder flags

NAME

compfloat – compare float

SYNOPSIS

int compfloat (float ****x*, float ****y***);**

ARGUMENTS

x	1st float to compare
y	2nd float to compare

DESCRIPTION

Return -1 if x is smaller than y, else 1

NAME

convl12 – converts a sensor-generic level-1b record to level-2

SYNOPSIS

int convl12 (l1str ****l1rec*, l2str ****l2rec***, int** *spix***, int** *epix***, int** *il***, instr ****input***);**

ARGUMENTS

l1rec	level-1b record
l2rec	level-2 record
spix	starting pixel
epix	ending pixel
il	current line number
input	user input parametes

DESCRIPTION

This function is the main driver for the conversion of a level-1b record into a level-2 record. Each record consists of a single scan line.

AUTHOR

B. A. Franz, GSC, SIMBIOS Project, March 1998

NAME

cpl1rec – copy level-1 structure

SYNOPSIS

void cpl1rec (l1str ****new*, l1str ****old***);**

ARGUMENTS

new	new level-1 record
old	old level-1 record

DESCRIPTION

Copy the level-1 structure. The coller is responsible for having previously allocating space (for both).

NAME

fLTRmean – perform mean filter

SYNOPSIS

void fLTRmean (l1qstr ****l1que*, int** *nx***, int** *ny***, int** *ib***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
ib	band number
l1rec	filtered level-1b record

DESCRIPTION

Initializes filter control structure to sane values

NAME

fLTRmed – perform median filter

SYNOPSIS

void fLTRmed (l1qstr ****l1que*, int** *nx***, int** *ny***, int** *ib***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
ib	band number
l1rec	filtered level-1b record

DESCRIPTION

Initializes filter control structure to sane values

NAME

fLTmean – perform mean filter

SYNOPSIS

void fLTmean (l1qstr ****l1que*, int** *nx***, int** *ny***, int** *ib***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
ib	band number
l1rec	filtered level-1b record

DESCRIPTION

Initializes filter control structure to sane values

NAME

fLTmed – perform median filter

SYNOPSIS

void fLTmed (l1qstr ****l1que*, int** *nx***, int** *ny***, int** *ib***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
ib	band number
l1rec	filtered level-1b record

DESCRIPTION

Initializes filter control structure to sane values

NAME

fLTrat – perform band ratio filtering

SYNOPSIS

void fLTrat (l1qstr ****l1que*, int** *nx***, int** *ny***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
l1rec	filtered level-1b record

DESCRIPTION

Band ratio filtering a la Watson Gregg

NAME

fctl_init – initialize filter control

SYNOPSIS

void fctl_init (fctlstr * *fctl***);**

ARGUMENTS

fctl	filter control structure

DESCRIPTION

Initializes filter control structure to sane values

NAME

fctl_set – initialize filter to desired type

SYNOPSIS

int fctl_set (fctlstr * *fctl***, int** *npix***, char *** *fname***, int** *band***, int** *nx***, int** *ny***);**

ARGUMENTS

fctl	filter control structure
npix	number of pixels in a record
fname	name of filter
band	band selected for filtering
nx	width of filter (should be odd)
ny	height of filter (should be odd)

DESCRIPTION

Initializes filter control structure to sane values

NAME

fdilate – perform dilation filter

SYNOPSIS

void fdilate (l1qstr ****l1que*, int** *nx***, int** *ny***, int** *flag***, l1str ****l1rec***);**

ARGUMENTS

l1que	queue of level-1b records
nx	width of filter
ny	height of filter
flag	type of data to flag
l1rec	filtered level-1b record

DESCRIPTION

Initializes filter control structure to sane values

NAME

filehandle_init – initialize filehandle struct

SYNOPSIS

void filehandle_init (filehandle * *file***);**

ARGUMENTS

file filehandle structure

DESCRIPTION

Initalizes the filehandle structure to good starting values.

NAME

filesize – returns file length in bytes

SYNOPSIS

int filesize (const char * *filename***);**

ARGUMENTS

filename input filename

DESCRIPTION

returns file length in bytes

CALLED BY

readl1hdr_g

NAME

filter – filter data record

SYNOPSIS

void filter (fctlstr * *fctl*, **l1qstr ***_l1que_, **l1str ***_l1rec_**);**

ARGUMENTS

fctl	filter control structure
l1que	queue of level-1b records
l1rec	level-1b record to filter

DESCRIPTION

Initializes filter control structure to sane values

NAME

free_l1 – frees l1str data

SYNOPSIS

void free_l1 (l1str ****l1rec*);**

ARGUMENTS

l1rec	level-1 data record

DESCRIPTION

Frees memory allocated by **alloc_l1**

CALLED BY

free_l1q

NAME

fresnel – compute fresnel reflectance

SYNOPSIS

float fresnel (float *theta*, **float** *rel_index*);

ARGUMENTS

theta	incident angle (radians)
rel_index	relative index of refraction (n2/n1)

DESCRIPTION

calculates the fresnel reflectance given the relative refractive index between two medium and the incident angle

NAME

get_N_particles – estimate the number of small and large particles

SYNOPSIS

void get_N_particles (l2str ****l2rec*, int** *recnum***, int** *band***, float ****Nsmall***, float ****Nlarge***);**

ARGUMENTS

l2rec	level-2 record
recnum	current record number
band	band number
Nsmall	number of small particles at desired band number
Nlarge	number of large particles at desired band number

DESCRIPTION

This routine uses a algorithm from Vladimir Haltrin to compute the total number of small particles and large particles based on the backscattering coefficient. The backscattering coefficient is computed using Arnone's method.

RETURNS

0 for success, otherwise, failure

NAME

get_aer_index – get aerosol index

SYNOPSIS

void get_aer_index (l1str ****l1rec*);**

ARGUMENTS

l1rec	level-1b record

DESCRIPTION

Gets the aerosol index for each pixel in the l1re

CALLS

aero_indx

NAME

get_angstrom – get angstrom data

SYNOPSIS

void get_angstrom (l2str ****l2rec*, int** *band***, float ****angst***);**

ARGUMENTS

l2rec	level-2 data record
band	desired band number
angst	angstrom coefficient

DESCRIPTION

Computes the angstrom coefficient relative to band 8 from models.

ALGORITHM

M. Wang

AUTHOR

B. A. Franz, SAIC GSC, SIMBIOS Project, 4 Aug 1999

NAME

get_carder – run carder desired algorithm

SYNOPSIS

int get_carder (double ***_rrs_, double ***_atot_**, double ***_aph_**, double ***_adg_**, double ***_bb_**, double ***_chl_**, double ***_agdef_**, double ***_aphdef_**, double ***_agmod_**, double ***_aphmod_**, double ***_s_**, int** _iter_s_**, short *** _flags_**);**

ARGUMENTS

rrs	remote sensing reflectance in nm for first six SeaWiFS wavelengths
atot	output total absorption spectrum (1/m)
aph	output phytoplankton absorption spectrum (1/m)
adg	output CDOM absorption spectrum (1/m)
bb	output backscattering spectrum (1/m)
chl	output chlorophyl-a (mg/m^3)
agdef	output default adg400
aphdef	output default aph675
agmod	output modeled adg400
aphmod	output modeled aph675
s	output spectral value
iter_s	iterate on s flag
flags	carder flags

DESCRIPTION

get_carder provides a single entry point to the various Carder algorithms.

CALLED BY

carder, **getadg412_case1**

CALLS

carder_v1_3, **carder_v1_33**, **carder_v1_4**

NAME

get_depth – get depth data

SYNOPSIS

void get_depth (l2str ****l2rec*, uint8 ****depth***);**

ARGUMENTS

l2rec	level-2 data record
depth	water depth index

DESCRIPTION

Computes water depth index for each pixel in data record.
0 = missing data
5 = shallow water (<~5m)
20 = medium water (~5m - ~20m)
100 = deep water
105 = 5 + pigment
120 = 20 + pigment
200 = 100 + pigment
240 = clouds
250 = land

ALGORITHM

R. A. Stumpf, NOAA

AUTHOR

B. A. Franz, SAIC GSC, SIMBIOS Project, 9 July 1999

NAME

get_k490 – get diffuse attenation at 490nm

SYNOPSIS

void get_k490 (l2str ***_l2rec_, float ***_k490_**);**

ARGUMENTS

l2rec	level-2 data record
k490	diffuse attenuation at 490 nm

DESCRIPTION

Computes diffuse attenuation at 490 nm for each pixel in data record.

ALGORITHM

J. Mueller

AUTHOR

B. A. Franz, SAIC GSC, SIMBIOS Project, 9 July 1999

NAME

get_k532 – get diffuse attenuation at 532nm

SYNOPSIS

void get_k532 (l2str ****l2rec*, int** *flag***, float ****k532***);**

ARGUMENTS

l2rec	level-2 data record
flag	1 - attenuation length, else attenuation
k532	diffuse attenuation at 532 nm

DESCRIPTION

Computes diffuse attenuation at 532 nm for each pixel in data record.

ALGORITHM

J. Mueller/Austin&Petzold

NAME

get_mos_N_particles – estimate the number of small and large particles

SYNOPSIS

void get_mos_N_particles (l2str ****l2rec*, int** *recnum***, int** *band***, float ****Nsmall***, float ****Nlarge***);**

ARGUMENTS

l2rec	level-2 record
recnum	current record number
band	band number
Nsmall	number of small particles at desired band number
Nlarge	number of large particles at desired band number

DESCRIPTION

Version of get_N_particles for SeaWiFS data.

NAME

get_swf_N_particles – estimate the number of small and large particles

SYNOPSIS

void get_swf_N_particles (l2str ****l2rec*, int** *recnum***, int** *band***, float ****Nsmall***, float ****Nlarge***);**

ARGUMENTS

l2rec	level-2 record
recnum	current record number
band	band number
Nsmall	number of small particles at desired band number
Nlarge	number of large particles at desired band number

DESCRIPTION

Version of get_N_particles for SeaWiFS data.

RETURNS

0 for success, otherwise, failure

NAME

rdfilter – read filter data

SYNOPSIS

int rdfilter (char * *file***, int** *npix***, fctlstr *** *fctl***);**

ARGUMENTS

file	name of filter file to read
npix	number of pixels in image
fctl	filter control structure

DESCRIPTION

Initializes filter control structure to sane values